Master's Thesis

## Master Degree in Energy Engineering

# PV plants grid integration analysis

**MEMORY**


**Autor:** Xavier Bautista
**Director:** Eduardo Prieto
**Ponent:** María Camino

**Convocatòria:** April 2020


Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona

**Abstract**

This project explains how to simulate a real photovoltaic power plant with its surrounding atmospheric conditions. It aims to make a simulation as close as possible to a real PV power plant functioning. However, this simulation has computer limitations due to the computer performance characteristics. Therefore, some simplifications have been made to release computational processing problems.

The simulation takes into account several data from different facility manufacturers in order to face the real generation for a randomized weather conditions. As the most focusing part is on the DC side the most accurate results are given for the DC side. Therefore, manufacturer variations can be controlled very well and variations among different manufacturers can be evaluated for the same weather conditions. This precision allows knowing which equipment has a better behavior for different test conditions.

It is proposed a simple voltage source converter to transform the DC power into AC power. The required would be the electrical data given by the manufacturer.

It is explained what steps have to be taken throughout the simulation construction and the processes susceptible to become a complaining problem or warning advices that slow down the entire simulation.

To do such a simulation many equations and algorithms are given throughout the document to transform input weather data to grid power injected. Moreover, during the process more intermediate outputs such as irradiance on the module field or DC power are obtained.

Finally, a general simulation is performed to show the whole simulation functioning. It is shown the traveling cloud, how the cloud is affecting directly the DC power and, consequently, the AC power. Furthermore, by using the simplified simulation is shown how the cloud affects the string of modules you are interested to focus on.

The conclusion is that the simulation results useful to make this kind of analysis. It could also be useful to those manufacturers that are interested in generating simulated inputs to their converter models and evaluate performances. All in all, it can be used in some other ways that help another kind of investigation. On the other hand, it needs a better debugging process to make it a little bit faster and, consequently, useful.

# Summary

# Figures

# Tables

# *1* Glossary

PV: Photovoltaic

STC: Standard test conditions

NOCT: Nominal operating cell temperature

PLL: Phase locked loop

CCL: Control current loop

RC: Reference computation

MPPT: Maximum power point tracking

P&O: Perturb and observe algorithm

IC: Incremental conductance

IAC: Adaptive Incremental conductance

IAIC: Improved adaptive incremental conductance

AC: Alternating current

DC: Direct current

IGBT: Insulated Gate Bipolar Transistor

VSC: Voltage source converter

# 2 Introduction

This project addresses into a particular issue on photovoltaic power production making deep focus in the know-how of power generation by means of simulating an entire system by given inputs considered as real data.

The thesis describes a detailed process to evaluate the specific power output of a photovoltaic power plant for given conditions and given installed capacity. The whole description is performed that results in a software simulation. This simulation is the result of the combination of the given information in Matlab and Simulink code.

## 2.1 Project targets

First of all, it is important to remark that it is faced particularly on distributed generation power plants. These kinds of power plants are considered from a few kilowatts installed to several megawatts installed [1]. In this project, it is considered from 100 kW to 10 MW.

The main target is to observe the dynamic effect produced in a power plant by means of variations in the irradiance behavior. This dynamic effect is used to evaluate the impact on the power production. In addition, it allows rethinking many different scenarios that could be produced around the world.

The second target is to feed the inverter designers with a tool they could use to generate input data to test their inverters however they are interested in. It is not a mandatory tool for just inverter designers but it could be used in whatever manner it could be thought. The concept is to present a complex model that outputs trustable data from confident input data.

The third target is to provide module manufacturers with a tool that can perform an accurate analysis of their module behavior. To clarify the purpose, this tool gives them the chance to analyze the module behavior for many different given weather conditions. This way, manufacturers can optimize their module designs adapting them to their given conditions. This optimization is thought in the way that they can manufacture modules focusing on environmental conditions. For example, it is known that for photovoltaic production it is desired to have the maximum irradiation but if it is required to install in a location that is usually cloudy it would made sense to manufacture modules specifically to that conditions [2]. Facts like spaces between cells, materials and cells connections are able to affect the generation in one way or another.

The last target is that allows evaluating complex MPPT systems. These MPPT systems do

not need to be the existing ones but they can be new implementations or enhanced MPPT algorithms coming from the existing ones. By inserting the MPPT algorithm the impact could be evaluated and trustable conclusions would be achieved.

## 2.2 Project scope

The studied case is a particular case based on a power plant of approximately 3 MW of peak power. It consists of 9180 modules connected in strings of 30 modules, resulting in 306 strings parallel connections in the inverter.

Its layout configuration consists on trackers separated with a pitch of 5 meters. Each tracker has been customized to be a large tracker. Therefore, if considers an amount of 51 trackers of 6 strings in landscape configuration. For this specific case, any specific manufacturer has been considered due to the fact that irradiance is given as input data, so it is considered a randomly tracker structure.

Lastly, it considers a central inverter with 1 MPPT input. This inverter is not considered as restrictive as it is been considered the rest of the approach because of the project focusing. Thereby, it is selected a beta inverter coming from General Electric central inverter. It is important to make some emphasize on the MPPT because it directly affects the power production and the power plant behavior. If the same problem approach has been considered with string inverters the programming code would consider the amount of MPPTs needed by inverter. Nevertheless, this particular case by using string inverters has not been considered in this analysis.

After presenting the problem, the theoretical framework is presented at full detail to face the problem approach at the maximum possible precision. Every concept is extremely developed in order to control the entire surrounding variables that could affect the final simulation in a given way.

Finally, the simulation is carried out in the solar power plant of 3 MW of peak power with modules of the manufacturer Canadian Solar. Each module has a power capacity of 330 Wp, CS3U-330P. The power plant also considers an ideal inverter based on General Electric LV5+ 1560 model features. It is important to know that the inverter is not fully detailed because the main focus is in the module field part. However, it is considered essential the power conversion due to the fact that all power plants are connected to the grid, no one works in an isolated mode.

## 2.3 Computational tools

A Project like the presented one does not make sense at all without an accompanied software giving results about the explained concepts. Otherwise, theory derives to studied cases that are never argued and demonstrated. Software provides useful tools that reduce the complexity of many procedures.

The software utilized extensively in this project is MATLAB-SIMULINK. This software is really extended software among science community. It provides a huge variety of tools that allow the programmer to study any particular case it is interested to study and it is applied in many different fields, not only electrical cases.

Martlab and Simulink count on enormous scientific community that shares other information that is relevant to other studies or it can be used in other ways. It also has a important helpful content that help users to solve technical issues in case of necessity [3].

As the software used in the project is a really powerful simulation tool, it is been stressed to its limits in order to see what is the maximum definition of the input variables and output results that the software is available to provide.

For this reason many trials are performed to achieve the maximum resolution. To do so, an iterative process of the most complex model is been performed until the simulation runs without huge computing problems.

# 3 Simulation target

The thesis aims to make an entire simulation of a power plant with the maximum detail definition of the model as possible of this power plant configuration.

## 3.1 First approach

The first model approach that has been tested is the full detailed power plant where all the existing facility equipments are considered at its maximum definition.

The module is understood as detailed as the manufacturer is defining it in the datasheet. It means that the module is understood as three different solar cells that are wired in series and having bypass diodes in parallel as seen in Figure 1.

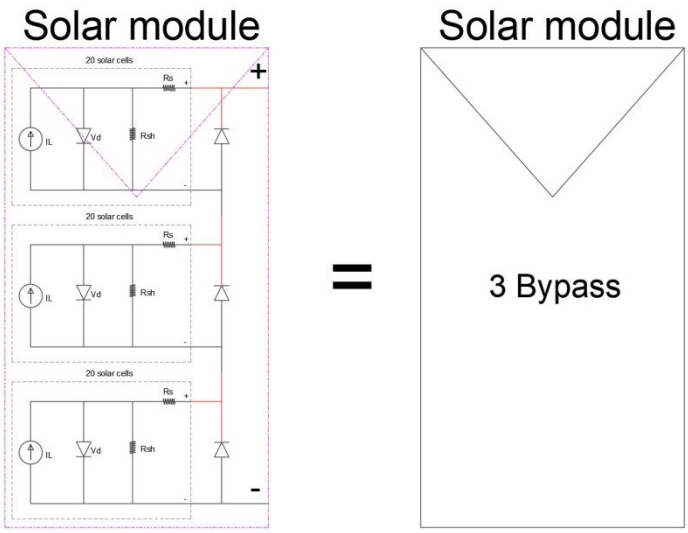


*Figure 1 Detailed solar module*

The model consists on wiring as many solar modules as shown in Figure 1 to an MPPT algorithm device that evaluates the maximum power point for the whole power plant.

The last configuration is related to VSC which is configured as one device according to the manufacturer features.

The main concept is to have the whole assembly as shown in Figure 2.

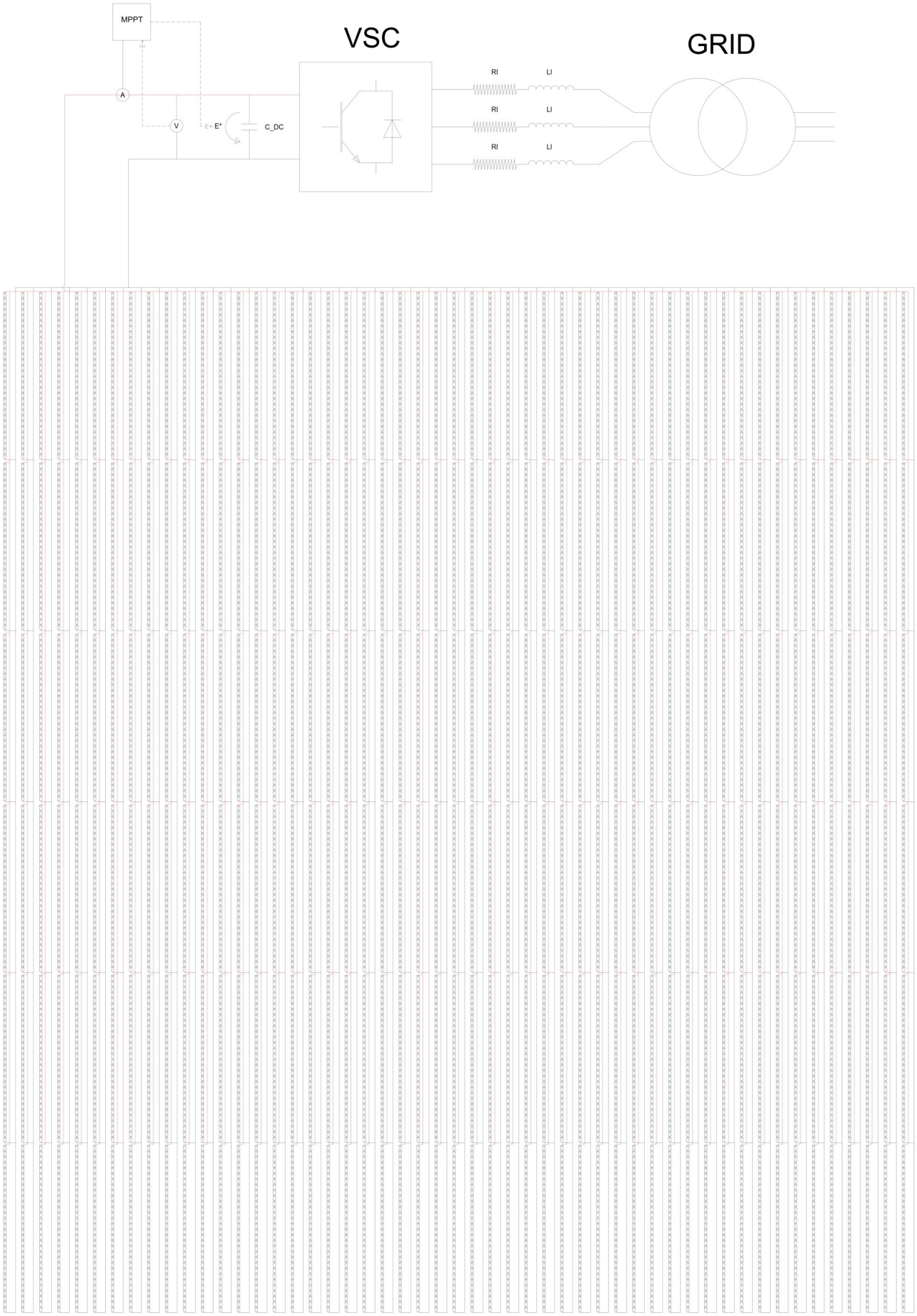


*Figure 2 Full detailed simulation*

This assembly gives the whole information in a full detailed manner. It can be analyzed perfectly every variable or parameter it is desired without any kind of simplification or assumption.

Nevertheless, when this configuration is intended to be programmed Simulink does not hold the entire simulation, it is so detailed the configuration that not even a 5 % of its configuration can be performed. It results in an absolutely system crush.

The maximum achieved configuration with this full detail configuration is about 100 kWp which is not even close to the expected for the case study. The generated Simulink file has an enormous amount of calculations that are not fast to be managed by the own program. If it was feasible to generate the whole configuration it would have taken too much time to compile and thus run the program for the complete simulation. However, the simulation crushes before even generation the PV configuration.

For this reason this first approach scenario has to be dismissed which means that a full detail simulation will never be achieved.

## 3.2 Second approach

This second model scenario has some simplifications compared with the previous model.

As the main problem of the previous model is produced because of the amount of modules considered and its corresponding detail, in this model the approach has been considered an important simplification compared with the previous one.

In this case a string is generated as an aggregated module. The module is been scaled to be equivalent as the amount of modules connected per string that were considered in the previous model. Moreover, this giant model has no bypass diodes in its internal configuration. Therefore, its internal configuration is equivalent to Figure 3.

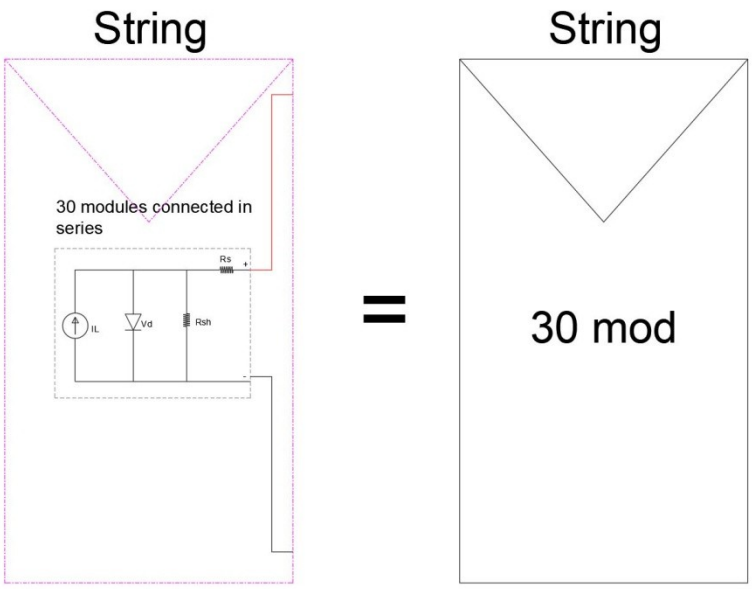


*Figure 3 String simplification*

The other equipment such as the MPPT and the VSC remain in the same configuration that was proposed in the first approach.

After considering this important simplification a working file is achieved. The file weight is about 2 Mb and has no problems when it is run. Figure 4 gives a clear view of the adapted configuration if it takes into account the simplification of Figure 3.

As the intention of the simulation is to provide a full detail simulation of the overall behavior of the power plant another model is created to analyze in a deeper detail the behavior of the strings. This new file allows one to know in a deeper accuracy the string functioning for a chosen string.

This new model is fed with the input irradiance and the voltage obtained from the main simulation where the MPPT has evaluated the optimal voltage for the maximum power production.

Figure 5 shows the detailed configuration of the chosen string. It is important to remark that this model does not count with the VSC because it is considered as an extra tool to make a deeper focus on the string behavior.

The importance of the simplification is that with a lower detail the model does not crash as it crushes in the first approach.

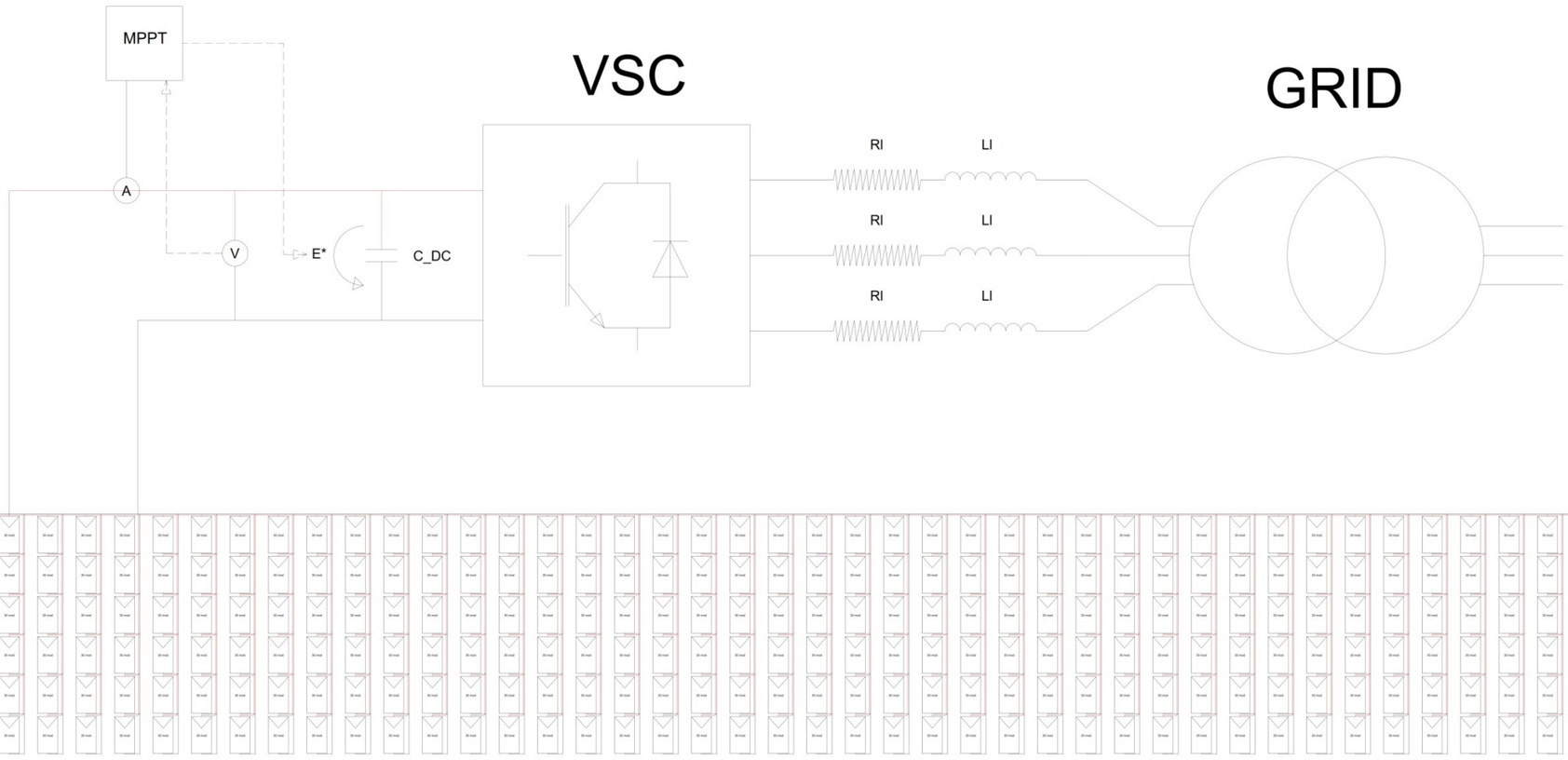


*Figure 4 Simplified simulation*

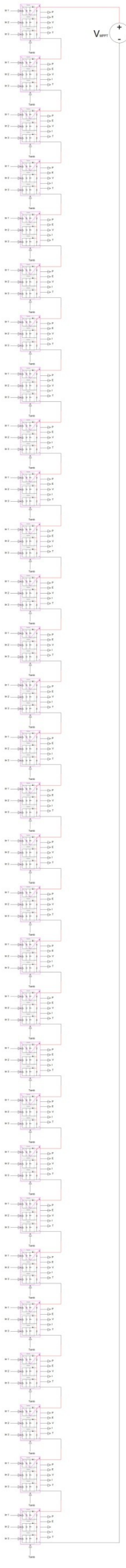


*Figure 5 Detailed string simulation*

# 4 PV theoretical framework

In this section, it is introduced the theoretical framework for the modeling techniques applied in this thesis. First, it is described the equations needed to model a PV panel. This thesis takes into account real data from a module manufacturer. Nevertheless, some information is not shared in their spreadsheets. For those cases, secondary input data will be assumed to achieve the main module parameters.

It is important to know that the scope of this part is focused from the electric point of view which means that it will not be taken into account the real science behind the PV module. The PV module will be understood as electric elements.

## 4.1 Module as electric elements

The main element in a PV power plant is the PV module which transforms the energy from the solar radiation, photons, into electrical power. They consist in two selective contacts and an absorber. In the particular case of the polycrystalline solar cells the selective contacts are the p-type and n-type regions. Then, the absorber is the depletion zone located between these selective contacts. In real dimension the depletion zone is a really thin layer that absorbs the photons to release an electron. The function of the p-type and n-type regions is just to filter the direction of this electron released.

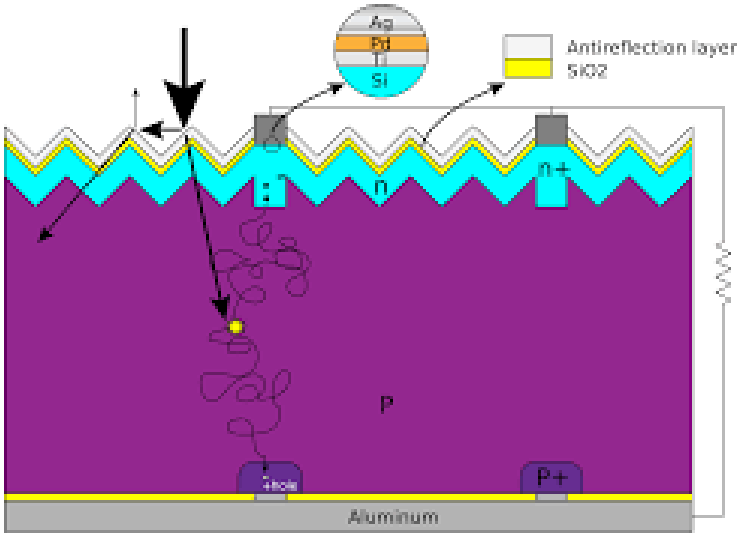


*Figure 6 Cross Section of a typical crystalline solar cell*

The commercial modules are mainly manufactured with polycrystalline or mono-crystalline

structures. There exist many different companies manufacturing this kind of commercial photovoltaic panels, so the chosen module is a module from on the TIER 1 most sold manufacturers among the world [4].

### 4.1.1 PV model

A PV solar call is understood as junction of some electrical elements. A current source, several diodes, a series resistance and a shunt resistance. It is important to remark that a real solar cell consist in many diodes but in order to make the model as simple as possible just one diode is considered per each PV module [5], [6].

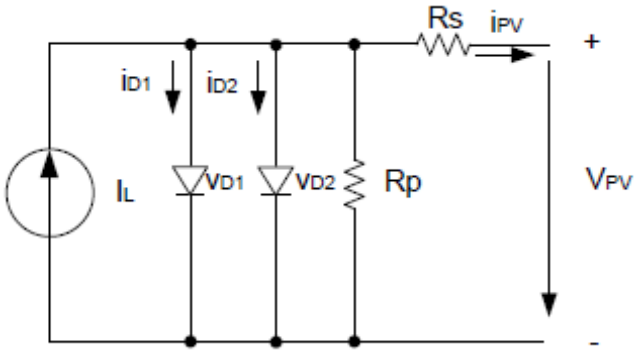


*Figure 7 Solar cell electrical model*

By knowing that the association of several solar cells can be understood as a simplified solar module as shown in Figure 8. The following equations for a general solar module are considered:

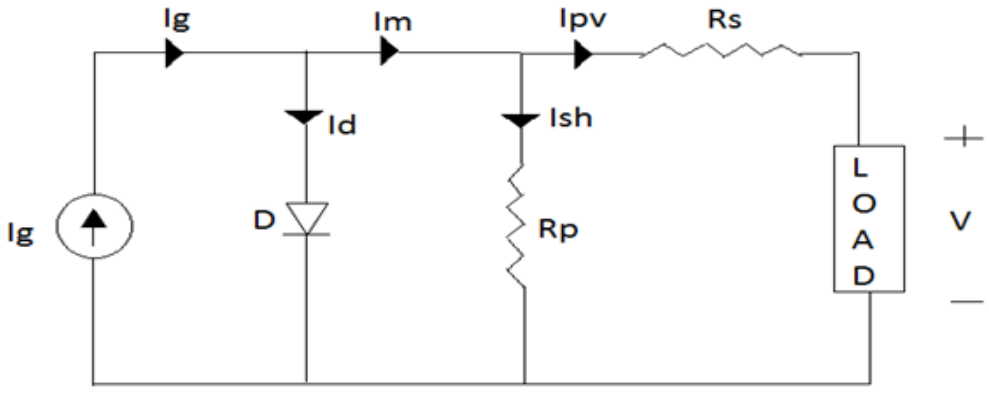


*Figure 8 Electric model of solar panel*

$$I_{pv} = I_m - I_{sh} \quad (1)$$

$$I_{sh} = \frac{V + I_{pv}\, R_s}{R_p} \quad (2)$$

$$I_m = I_g - I_d \quad (3)$$

$$I_g = \left(I_{g,n} + K_I \Delta T\right) \frac{G}{G_n} \quad (4)$$

$$\Delta T = T_{cell} - 25 \quad (5)$$

$$T_{cell} = T_{amb} + (NOCT - 20)\, \frac{G}{G_n} \quad (6)$$

$$I_d = I_0 \left(\exp\left(\frac{V + K_v\, \Delta T}{n V_t}\right) - 1\right) \quad (7)$$

$$I_0 = \frac{I_{sc,n} + K_I\, \Delta T}{\exp\left(\frac{V_{oc,n} + K_v\, \Delta T}{n\, V_t}\right) - 1} \quad (8)$$

$$v_T = \frac{kT}{q} \quad (9)$$

$$I_{pv} = I_g - I_d + I_{sh} \quad (10)$$

Where:

- $I_{pv}$ is the generated current going out of the pv module (A)
- $I_d$ is the diode current that represents the recombination current (A)
- $I_{sh}$ is the current produced because of the parasitic losses (A)
- $I_g$ is the generated photocurrent by the pv module (A)
- $R_p$ is the resistance associated to the parasitic losses (Ω)
- $R_s$ is the resistance associated to the leakage current before leaving the solar module (Ω)
- G is the solar irradiation (W/m²)

- $G_n$ is the nominal irradiation at NOCT conditions (W/m²)
- NOCT is the nominal operating cell temperature at NOMT conditions (°C)
- $T_{cell}$ is the cell temperature (°C)
- $I_{g,n}$ is the nominal current generated by the solar module (A)
- $K_I$ is the current temperature coefficient (%/°C)
- $K_v$ is the current temperature coefficient (%/°C)
- $I_o$ is the diode saturation current (A)
- $V_t$ is the thermal voltage (V)
- k is the Boltzman constant (J/K)
- q is the electron charge (C)
- n is the diode ideality factor

### 4.1.2 Manufacturer module configuration

Knowing that a solar module is a junction of many solar cells the model to be designed will be a simplified junction of many solar cells. The amount of solar cells to be considered in the model will depend on the bypass diodes of the own module. The configuration will be considered depending on the manufacturer but generally each module is composed of 3 bypass diodes that allow reducing the losses in case of shadowing [7].

To make a reasonable model is it considered one recombination current diode instead of some recombination current diodes per each solar cell electric model, it makes the model simpler.

Therefore, each module will consist in as many solar panels as shown in Figure 8 as bypass diodes considered by the PV manufacturer.

To generate the correct modeling module the following variations on the equations are taken into account:

$$V_{oc,part} = \frac{V_{oc,mod}}{n_{diodes}} \quad (11)$$

$$n_{cells\ ,part} = \frac{n_{modcells}}{n_{diodes}} \quad (12)$$

$$R_{s,part} = \frac{R_s}{n_{diodes}} \quad (13)$$

$$R_{sh,part} = \frac{R_{sh}}{n_{diodes}} \quad (14)$$

$$\mathrm{v_T} = \frac{n_{cells\ ,part}\ \mathrm{k\,T}}{\mathrm{q}} \quad (15)$$

Where:

- $V_{oc,part}$ corresponds to the open circuit voltage for the solar part model(V)
- $n_{cells,part}$ corresponds to the number of cells for the solar part model (units)
- $n_{diodes}$ corresponds to the number of bypass diodes of the module (units)
- $R_{s,part}$ corresponds to the series resistance of the solar part module (Ω)
- $R_{sh,part}$ corresponds to the shunt resistance of the solar part module (Ω)

### 4.1.3 Simplified string configuration

In order to simplify the general simulation PV generation solar field, a whole string of 30 modules is also modeled as a string simplification.

This simplification consists on an association of 30 modules per string. Thus, some equation variations are considered from the general PV module allowing the string to be considered as a giant module without bypass diodes. It is important to take into account that this simplification will affect the real input irradiance because 90 bypass diodes are neglected.

$$\mathrm{v_T} = \frac{n_{cells}\ \mathrm{k\,T}}{\mathrm{q}} \quad (16)$$

$$\mathrm{n_{cells}} = \mathrm{n_{cells\ ,mod}}\ \mathrm{n_{mod}} \quad (17)$$

$$\mathrm{V_{OC,string}} = \mathrm{V_{OC,mod}}\ \mathrm{n_{mod}} \quad (18)$$

$$\mathrm{R_{s,string}} = \mathrm{R_{s,mod}}\ \mathrm{n_{mod}} \quad (19)$$

$$R_{sh,string} = R_{sh,mod}\ n_{mod} \quad (20)$$

Where:

- $V_{oc,string}$ corresponds to the open circuit voltage for the string(V)
- $n_{mod}$ corresponds to the number of solar modules per string (units)
- $n_{cells}$ corresponds to the number of cells in the string (units)
- $R_{s,string}$ corresponds to the series resistance of the solar part module (Ω)
- $R_{sh,string}$ corresponds to the shunt resistance of the solar part module (Ω)

In this particular case a mean value of the irradiation received by the whole string will be considered as irradiance input to the model.

## 4.2 Module association

This project addresses mainly two different module associations. The main reason is due to the evaluation cannot be executed all at once because of computing limitations.

As this is a major reason, a very detailed association has been programmed in order to consider all elements that are present in the string.

The other association consists in a simplified association that minimizes the elements needed to process the simulation.

### 4.2.1 Simplified string association

This association consists on following the statements set in 4.1.3 Simplified string configuration and simplify a connection of 30 modules per string.

After that, there are considered 306 equal strings elements connected in parallel. This association aims to emulate the connection of the real DC side of the power plant.

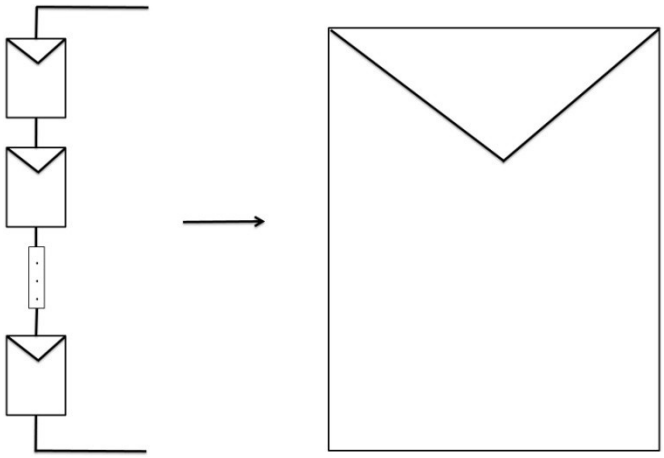

*Figure 9 String simplification*

This configuration is designed to evaluate the power generation in the power plant. Nevertheless, it just allows knowing the power generation in the one string. It is not possible to evaluate at fully detail what it is happening at every part of the module.

### 4.2.2 Detailed string association

This association is configured in order to minimize the distortion created by the previous association 4.2.2 Detailed string association. As it is not possible to evaluate at fully detail all the module production and affectations, this configuration is suggested so as to characterize one desired string.

The detailed string relies on 30 modules connected in series as it is shown in Figure 10. Every module is configured as stated in 4.1.2 Manufacturer module configuration by considering every bypass diode and 3 simplified solar cells per module.

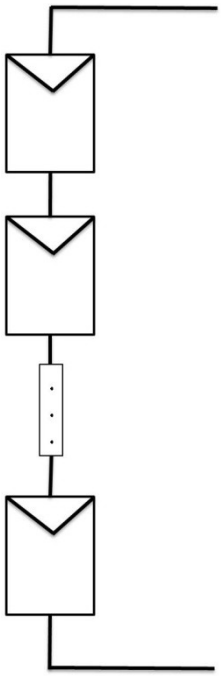

*Figure 10 Module association in series*

This association gives a more accurate result because it provides the exact response for every module part.

## 4.3 MPPT Algorithm

The main functionality of this part is to track the maximum power generation for the given atmospheric and irradiation conditions.

There exist many different procedures to reach the maximum power point but they have different implementation algorithms and results. The most used algorithms are the incremental conductance (IC), the perturb and observe (P&O), the variants of the perturb and observe algorithm, the adaptive incremental conductance (IAC), and the improved adaptive incremental conductance (IAIC) [8]–[11]. All these algorithms are useful for the PV implementation in an overall simulation.

However, there are some algorithms that provide more accurate results than others. In Table 1, it is shown the most used algorithms and efficient are evaluated to demonstrate which one has the most overall efficiency.

| Method | Static Efficiency | Dynamic Efficiency | Overall Efficiency |
|---|---|---|---|
| **CLASSIC P&O** | 86,43% | 89,95% | **88,09%** |
| **SC P&O** | 80,41% | 28,37% | **54,39%** |
| **P&O INTERPOLATION** | 89,00% | 91,30% | **91,15%** |
| **PI P&O** | 65,19% | 77,59% | **71,39%** |
| **CLASSIC IC** | 83,07% | 89,89% | **86,48%** |
| **AIC** | 88,13% | 91,13% | **89,63%** |
| **IAIC** | 87,82% | 91,18% | **89,50%** |

*Table 1 MPPT Algorithms efficiency* [9]

Considering the evaluations processed in different studies the classic perturb and observe algorithms is suggested to minimize the algorithm without affecting qualitatively the results.

This algorithm is constantly generating variations in the voltage in order to create differences between the current power measurement and the previous power measurement. By evaluating how the power and the voltage change the following voltage variation step is performed.

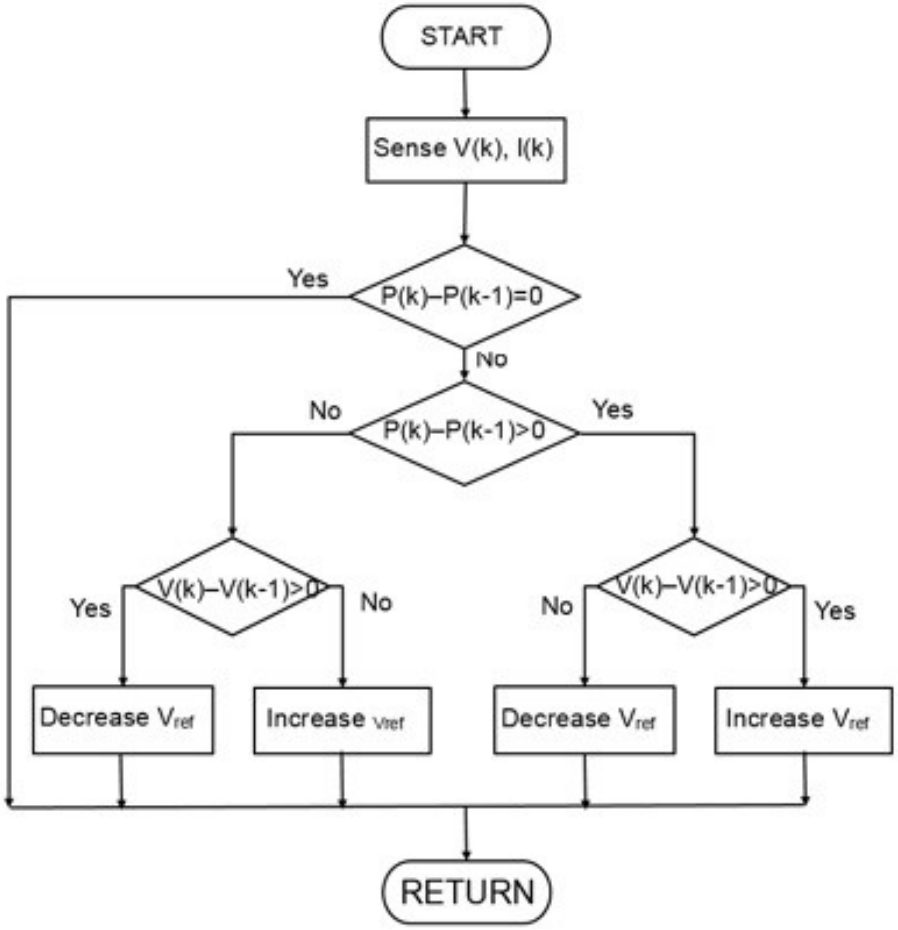


*Figure 11 P&O MPPT algorithm [9]*

Additionally, the step voltage variation has to be established in the algorithm so as to make the power evaluations. The smaller the step voltage variation the bigger the algorithm accuracy will be. On the other hand, the bigger the step voltage variation the smaller the accuracy will be.

# 5 Grid-connected voltage source converter

In this section it is described the theoretical framework for the modeling inverter part connected to the grid. This section consists on the coupling of several elements in order to transform the DC power coming from the modules and the MPPT to AC grid power.

The main elements involved in the control part are the Phase Locked Loop (PLL), the Reference Computation (RC), the Current Control Loop(CCL).

The process to convert the energy coming from the DC side to the AC side is shown in Figure 12.

It starts by reading the currents and voltages on grid side. The first process is applying the park transformation to the voltage measurements converting them from the abc frame to the αβ0 frame. After that, these lectures are processed through the phase look loop which establishes the current phase. Then, applying the same park transformation process to the current measurement and the obtained phase in the previous phase it is found the currents in the qd0 frame which is not a time domain frame anymore.

Before entering the currents of the grid side to the current loop another process has been started in parallel with the phase looked loop. This time is the reference computation current that is coming from the DC side of the inverter. It started by introducing the reactive energy coming from the grid and the voltage applied in the DC side in the current time step and the previous time step. The reference computation allows knowing which should be the reference currents of the grid side.

After these two processes both currents are introduced in the current control loop and the voltages in the qd0 frame are obtained.

Finally, these voltages are introduced in the voltage modulation which could response on two different ways. The first one would be by ordering the commutation of the IGBTs of the inverter and the second one as a decoupling of the qd0 frame to abc frame applied as AC voltage sources.

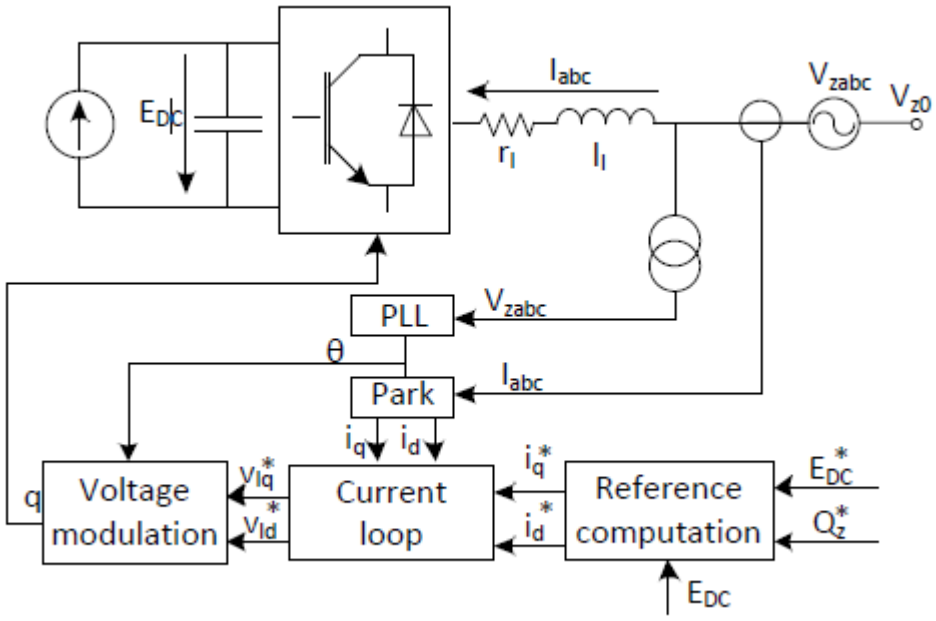


*Figure 12 Grid converter control scheme* [7]

This project discards the commutation process because of its slower simulation speed, being the voltage modulation by commutations neglected in this topic. All in all, this part will be simplified as direct AC voltage sources as explained before.

## 5.1 Phase locked loop

The phase locked loop is used to determine the angle and the angular velocity of the electrical network.

It mainly consists in a feedback of the d-axis voltage filtered through a PI controller. At the output of the controller it is achieved the angular velocity of the electrical network. Finally, integrating this angular velocity the phase is known, as well.

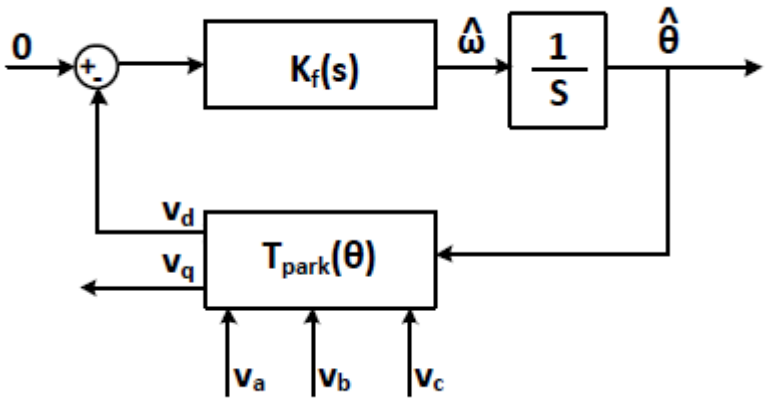


*Figure 13 Phase Lock Loop diagram Source:* [7]

The PI controller for the phase lock loop is defined as

$$K_f(s) = K_p\left(\frac{\left(\frac{1}{\tau_{PLL}}\right)+s}{s}\right) \quad (21)$$

$$w_n = \sqrt{\frac{K_p\ E_m}{\tau_{PLL}}} \quad (22)$$

$$\varepsilon = \sqrt{\frac{\tau_{PLL}\ K_p\ E_m}{2}} \quad (23)$$

Where:

- $\tau_{PLL}$ corresponds to PLL time constant (s)
- $E_m$ corresponds to admitted peak voltage value (V)
- $W_n$ corresponds electrical angular velocity (rad/s)
- $\varepsilon$ corresponds to the damping ratio

## 5.2 Reference computation

This section explains how to obtain the reference electrical currents as inputs to the control current loop.

The current references are obtained from the instantaneous power theory [12]. As explained in Annex B Park transformation, it is needed to work with constant quantities due to the easiest processing. Therefore, the instantaneous currents and voltages have to be transformed from the abc frame to the qd0 frame. Where the instantaneous currents and voltages are expressed as

$$x_a(t) = \sqrt{2}\, X\, cos(w\, t + \emptyset) \tag{24}$$

$$x_b(t) = \sqrt{2}\, X\, cos\left(w\, t + \emptyset - \left(\frac{2\pi}{3}\right)\right) \tag{25}$$

$$x_c(t) = \sqrt{2}\, X\, cos\left(w\, t + \emptyset + \left(\frac{2\pi}{3}\right)\right) \tag{26}$$

After transforming them by using the Park transformation it is obtained

$$x_\alpha = \sqrt{2}\, X\, cos(w\, t + \emptyset) \tag{27}$$

$$x_\beta = -\sqrt{2}\, X\, sin(w\, t + \emptyset) \tag{28}$$

$$x_0 = 0 \tag{29}$$

Where:

- X corresponds to the voltage or current measurement
- w corresponds to the electric grid frequency (rad/s)
- ∅ corresponds to the phase angle shift (rad)

The power expression is obtained from de deduction of the three phase power expression in the abc frame.

$$\underline{S} = P + j\underline{Q} = 3\, V^{\alpha\beta}\; I^{\alpha\beta *} = 3\left(\frac{v_\alpha - j v_\beta}{\sqrt{2}}\right)\left(\frac{i_\alpha + j i_\beta}{\sqrt{2}}\right) \tag{30}$$

Recombining the equations it is decoupled the active power from the reactive power

$$P = \frac{3}{2}(v_\alpha\, i_\alpha + v_\beta\, i_\beta) \tag{31}$$

$$Q = \frac{3}{2}(v_\alpha\, i_\beta + v_\beta\, i_\alpha) \tag{32}$$

By replacing the αβ0 frame constants to qd0 constants and isolating the currents the expression it is finally achieved the reference currents to introduce into the current control loop. Although, it exists a d-axis voltage the phase looked loop ensures that this component remains always at the value of 0.

Moreover, the power measurement reference calculation has to be computed from the grid side. Therefore, considering these last statements it is obtained the following reference currents

$$i_q^* = \frac{2}{3}\frac{P^*}{v_{zq}} \tag{33}$$

$$i_d^* = \frac{2}{3}\frac{Q^*}{v_{zq}} \tag{34}$$

Finally, the reference currents remain dependent on the reactive power required from the grid and the power generation of the DC side of the power plant. The voltage value is coming from the park transformation calculated for the voltage grid.

## 5.3 DC Voltage Regulator

The DC voltage regulator is required to know the active power generated from the DC side of the power plant. The DC voltage will provide the reference current for the control current loop.

The scheme adopted to control the DC bus is sketched in Figure 14, where is seen that the control relies in a feed-forward scheme to control the energy stored in the capacitor.

The active power reference depends on the injected power from the capacitor and the measured power before the capacitor.

$$P^* = P_C^* + P_{DC}^* \tag{35}$$

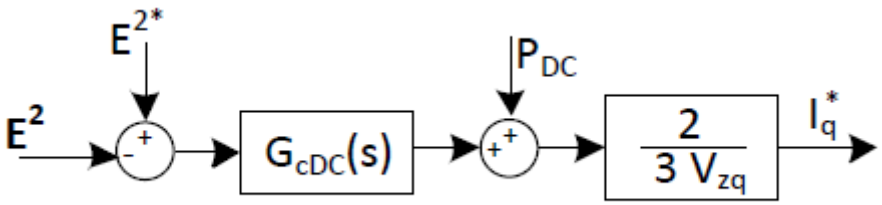


*Figure 14 DC Voltage Regulator Source:*[7]

The capacitor power expressed in the Laplace domain is used to start the control process

$$P_C(s) = \frac{1}{2}\, s\, CW(s) \tag{36}$$

A PI controller is implemented to control the feed-forward control. Therefore, the controller implemented is computed as

$$G_{c\,DC}(s) = K_{p\,DC} + \frac{K_{i\,DC}}{s} \tag{37}$$

The consequent controller gains are calculated as

$$K_{p\,DC} = C\, \varepsilon_E\, w_E \tag{38}$$

$$K_{i\,DC} = \frac{C\, w_E^2}{2} \tag{39}$$

Where:

- C is the capacitor capacity (F)
- $W_E$ is the angular velocity (rad/s)
- $\varepsilon_E$ is the desired damping ratio of the DC voltage loop

Whereas the DC power coming before the capacitor is implemented using a simplified model obtained by decoupling the DC and the AC parts of the converter.

$$P_{DC} = I_{DCl}\, E_{DC} \tag{40}$$

The power generated in the power plant is exchanged with the grid side. Therefore, it is assumed that all the power generated in DC is equivalent to the power converted to AC. Figure 15 shows how this power calculation is implemented in current mode.

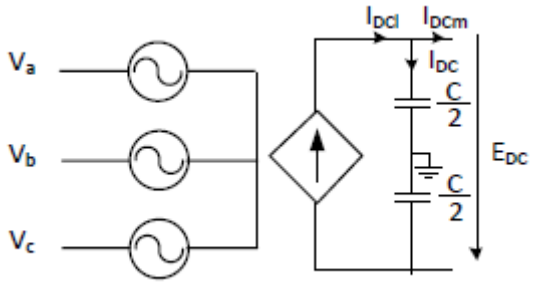


*Figure 15 Voltage Source converter model* [7]

## 5.4 Current Control Loop

The current control loop is the process in charge of providing the output voltages of the voltage source converter. To do the process it has to receive the reference currents and the measured currents from the grid.

First of all, it is needed to model the equivalent electrical circuit of the AC side which connects the voltage source converter with the grid side part, shown in Figure 16.

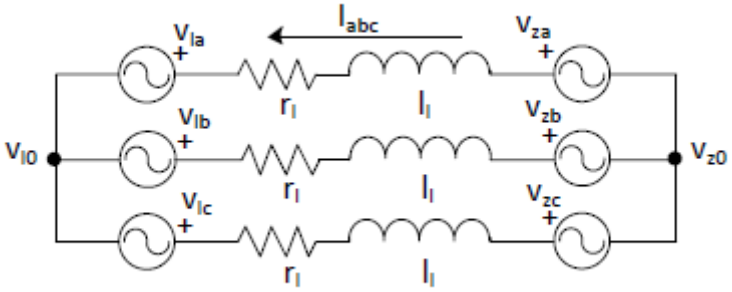


*Figure 16 Equivalent model of the AC side converter connected to the grid* [7]

The equivalent equation to Figure 16is

$$\begin{bmatrix} v_{za} \\ V_{zb} \\ V_{zc} \end{bmatrix} - \begin{bmatrix} v_{la} \\ V_{lb} \\ V_{lc} \end{bmatrix} - (v_{0l} - v_{0z}) \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} r_l & 0 & 0 \\ 0 & r_l & 0 \\ 0 & 0 & r_l \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \begin{bmatrix} l_l & 0 & 0 \\ 0 & l_l & 0 \\ 0 & 0 & l_l \end{bmatrix} \frac{d}{dt} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} \quad (41)$$

Where:

- $r_l$ is the inductance equivalent resistance of the inverter (Ω)

- $l_l$ is the inductance value of the inverter (H)
- $V_{za,zb,zc}$ are the three-phase instantaneous grid voltages in the abc frame (V)
- $V_{la,lb,lc}$ are the three-phase instantaneous converter voltages in the abc frame (V)
- $I_{a,b,c}$ are the three-phase instantaneous current in the abc frame (A)
- $V_{l0}$ is the neutral voltage converter (V)
- $V_{z0}$ is the neutral voltage grid (V)

Considering that neutrals are ground connected, Park transformation is applied to (41) and in the qd0 domain the current flowing in the $i_0$ variable is null it is expressed the equation as

$$\begin{bmatrix} v_{zq} \\ v_{zd} \end{bmatrix} - \begin{bmatrix} v_{lq} \\ v_{ld} \end{bmatrix} = \begin{bmatrix} r_l & l_l w_e \\ -l_l w_e & r_l \end{bmatrix} \begin{bmatrix} i_q \\ i_d \end{bmatrix} + \begin{bmatrix} l_l & 0 \\ 0 & l_l \end{bmatrix} \frac{d}{dt} \begin{bmatrix} i_q \\ i_d \end{bmatrix} \quad (42)$$

Where:

- $r_l$ is the inductance equivalent resistance of the inverter (Ω)
- $l_l$ is the inductance value of the inverter (H)
- $V_{zq,zd}$ are the grid voltages in the qd0 frame (V)
- $V_{lq,ld}$ are the converter voltages in the qd0 frame (V)
- $I_{q,d}$ are the currents in the qd0 frame (A)

Secondly, the components of the qd0 frame are decoupled related to the grid side or the converter side. The other condition coming from the phase locked loop is that the d-axis component on the grid side is considered null as considered in other sections.

$$\begin{bmatrix} v_{lq} \\ v_{ld} \end{bmatrix} = \begin{bmatrix} -v^{**}{}_{lq} + v_{zq} - l_l\, w_e\, i_{ld} \\ -v^{**}{}_{ld} + l_l\, w_e\, i_{lq} \end{bmatrix} \quad (43)$$

Where:

- $v_{lq,zq}$ are the voltages applied by the converter (V)
- $v^{**}{}_{iq,id}$ are the outputs of the current controller (V)

Substituting in the voltage equation (42) and applying Laplace transformation it is obtained

the transfer function between the controller voltages and current converters is obtained.

$$\frac{v^{**}{}_{lq}(s)}{i_q(s)} = \frac{1}{l_l\, s + r_l} \tag{44}$$

$$\frac{v^{**}{}_{ld}(s)}{i_d(s)} = \frac{1}{l_l\, s + r_l} \tag{45}$$

The controller designed uses the Internal Model Control technique [13].

$$G_{ciq}(s) = G_{cid}(s) =) \frac{K_p\, s + K_i}{s} \tag{46}$$

$$K_p = \frac{l_l}{\tau} \tag{47}$$

$$K_i = \frac{r_l}{\tau} \tag{48}$$

Where:

- $K_p$ and $K_i$ are constants
- $\tau$ is the closed loop time constant (s)

Finally, the overall control current loop will remain as sketched in Figure 17.

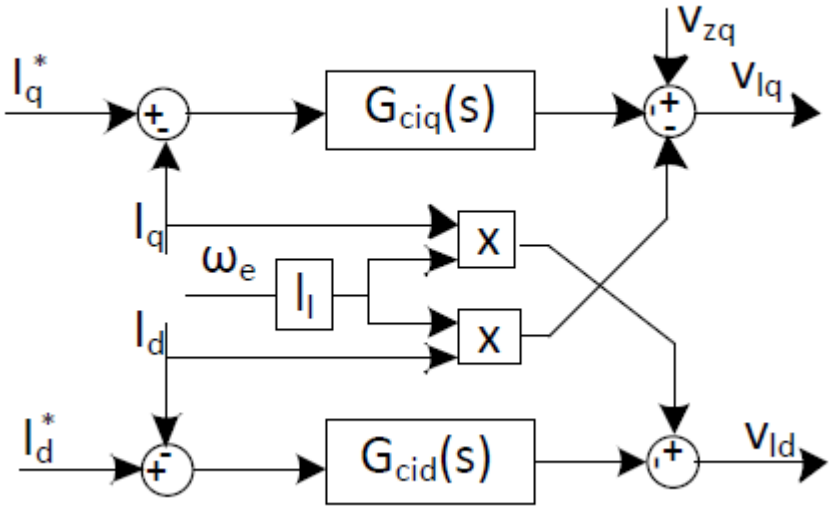


*Figure 17 Control current loop* [7]

# 6 Basic Cloud dynamics

This section introduces briefly the basic dynamics considered to simulate a cloud displacement. This topic does not address how clouds change their own shape duo to the wind effect and its transparency or any other climate effect that could produce slight variations on the irradiance effect or Albedo effect.

It is known that clouds absorb the direct irradiation coming from the sun but it is not calculated this absorption or the scattering effect of the cloud because this evaluation is not the core of the project.

The most important fact on the cloud dynamics is to know its location to consider given irradiance.

## 6.1 Cloud dynamic movement

Due to the random effect of wind and the real cloud movement several scenario could be considered in a 2D layout. Many mathematical paths could be described in order to see the real effect of the cloud on the module solar field, examples like sinusoidal, exponential, rotational…

However, the easiest case is considered so as to see how it affects the shadowing on the module solar field. A straight uniform movement is considered because it behaves as a quite normal cloud. The reason is because sky clouds usually go across in a quite linear path without deviating from a linear way.

Therefore, the linear equation from the basic dynamics is considered as the governor equation to indicate the path [14].

$$x = x_o + vt + at^2 \quad (49)$$

$$v = v_o + at \quad (50)$$

Where:

- $X_0$ is the initial position of the cloud (m)
- V is the cloud velocity (m/s)
- a is the acceleration (m/s²)

- t is the time variable (s)
- $v_0$ is the initial speed velocity

To simplify the cloud behavior it is considered a constant speed from the first starting step time. The reason is because it is desired to simulate the cloud as a constant moving element. To support this first fact acceleration will remain null so as to keep the cloud moving in a certain direction during the whole process.

Considering these arguments the cloud will behave the more similar as possible as it would behave in real life without creating really complex models.

## 6.2 Solar field shadowing

A solar field must be dimensioned at a given layout solar field. The concept is to perfectly know when the cloud is located upon the solar field.

To evaluate this synchronization it is needed to know a point location of the cloud in the sky positioning. Then, both layouts are overlapped in order to evaluate what modules are shadowed and what modules are not shadowed.

Mathematically, this problem is treated as several matrixes where every position of the matrix will determine a coordinate location in meters.

The cloud is dimensioned in whatever dimension it is required to dimension in the simulation. This matrix dimensions is understood as a field of ones and zeros. One means that there is cloud whereas zero means there is no cloud, this technique is used if it is required to evaluate stranger cloud shapes.

Then, the solar layout is sketched in a solar module field. This solar module field as the cloud field is dimensioned in zeros and ones. Where one means that there exist a solar module for that location and zero means that there is no solar module in that specified coordinate.

The solar field cloud is evaluated by considering one cloud point coordinate and applying (50). After that, the whole cloud is copied in the solar field matrix. As this cloud matrix does not exactly fit in the solar module field an interpolation must be considered. A complex algorithm evaluates the amount of cloud in per unit domain. Therefore, the cloud matrix location is not an integer matrix as the cloud dimensioning or the solar layout matrix.

$$\underline{SF}(x,y,t) = \underline{SF_0}(x,y) + \underline{C}(x,y,t) \qquad (51)$$

Where:

- $\underline{SF}$ is the output cloud matrix in the module solar field boundaries
- $\underline{SF_0}$ is the input solar field matrix in the module solar field boundaries
- $\underline{C}$ is the cloud matrix

The whole algorithm for the solar field evaluation is shown in the Annex C Cloud dynamics.

The output of (51) is a matrix of the solar field dimensions in values between zeros and ones. The value of each matrix position represents in per unit domain the amount of solar radiation that must be considered for the following calculation.

At this moment, the obtained result is representing the cloud at a given time upon the solar field considered layout.

## 6.3 Irradiation on the field

At this time, it is evaluated the real solar field irradiation. It is considered two different irradiances, one bigger than the other.

The bigger irradiance is the irradiance considered when there is no cloud in the sunlight direction, this irradiation has to be between the ordinary values of the earth irradiance values.

The smaller irradiance is irradiance produced when the cloud is crossing the sunlight between the sun and the module. This value will have to be smaller than the previous one because it simulates that the irradiance has dropped because of the cloud.

This model does not evaluate the density of the cloud that is why two irradiances are given. In a real case it would be measured with a pyranometer the irradiation without any clouds and the irradiation in a cloudy day.

By knowing this precondition it is formulated that

$$\underline{Irr_{field}}\,(x,y,t) = \left(\underline{Sf_{rad}}\,(x,y) - \underline{Sf}\,(x,y,t)\right)\,(irr_{maxi} - irr_{mini}) + irr_{mini} \qquad (52)$$

Where:

- $Irr_{field}$ is the irradiation matrix on the solar field (W/m²)

- $SF_{rad}$ is a matrix of ones in the same dimensions of the SF
- $Irr_{maxi}$ is the bigger irradiance considered (W/m²)
- $Irr_{mini}$ is the smaller irradiance considered (W/m²)

The (52) represents a linear regression that allows calculating accurately the value of the irradiation when the cloud is not perfectly fitted in the solar matrix field.

It has to be comprehended that the matrix $Sf_{rad}$ is given in ones because it represents total surface being irradiated by the sun. When the cloud is overlapped, the equation considers the subtraction of the cloud on the solar matrix.

At this moment, it is considered that the total irradiation that is reaching the solar field is perfectly known.

## 6.4 Irradiation on the modules

The last part consists in considering the module layout. Until this point it was only necessary the cloud dimensions, the surface field to be studied and irradiances to be applied.

This part only consists in locating the modules in a matrix of the same dimension as $SF_0$. This layout must consider the structure it is desired to implement. It could be considered two-axis tracking, one axis tracking or fixed structure.

$$\underline{Irr_{Tf}}(x,y,t) = \underline{Irr_{field}}\,(x,y,t)\,\underline{Tf}(x,y) \qquad (53)$$

Where:

- $Irr_{Tf}$ is the irradiance on the solar modules (W/m²)
- Tf is the matrix location of the modules

In this case the matrix location remains always constant. Besides, it consists on a matrix of ones and zeros. Ones means there exist a module in that coordinates and zero means that there not exist modules in that coordinate.

The output matrix is the irradiance that will be used as input for the solar modules in 4 PV theoretical framework.

# 7 Case study

## 7.1 Problem to be simulated

The simulation consists on gathering the necessary equipment to evaluate the power production of a certain amount of irradiance coming from the sun irradiation. Before simulating, the weather features and surrounding considerations are performed to generate a particular environment scenario.

This facility consists on a power plant of 3 MW of peak power, where peak power is the power coming from the PV modules. The number of PV modules installed is 9180 modules.

The configuration of the power plant is designed by groups of 30 modules per string. Hence, the power plant consists of 306 strings of 30 modules connected in series. Finally, these strings are connected in parallel at the entrance of the DC side of the VSC.

The reason to connect 30 modules per string is because is a really standard string measurement for PV trackers that allow the rounding of PV modules to be bought. Without forgetting that is a really common configuration that does not allow the open circuit voltage to reach the maximum survival voltage of the PV modules.

The facility considers a central inverter of 3 MW of nominal power, where nominal power is the power that the VSC can convert from DC to AC. Where, the network connection of this power plant is considered at low voltage.

The grid connection is considered ideal meaning that there is no strange perturbation, transient coming from the grid, reactive power to compensate and grid black out. These effects could affect in a direct way the functioning of the inverter but they are not considered in this simulation. Whereas the reactive power compensation is implemented.

It is been used a tracking structure on the solar field to install the PV modules. The power plant is composed of 51 trackers of 360 meters length.

Other equipment not involved directly on the power generation or the power conversion has been neglected because it does not focus on the problem to be evaluated.

| **Power plant equipment** | | **Unit** |
|---|---|---|
| **n° modules** | 9180 | u |
| **n° inverters** | 1 | u |

| | | |
|---|---|---|
| **n° trackers** | 51 | u |
| **n° strings** | 306 | u |
| **n° modules / string** | 30 | u |

*Table 2 Facility equipment*

The weather conditions to be analyzed do not consider any specific common weather case behavior of any particular location. The studied case considers some randomly input data from no specific location.

| **Weather conditions** | | **Units** |
|---|---|---|
| **Wind speed** | 20 | m/s |
| **Wind Direction** | 40 | ° |
| **Cloud dimensions** | 150x150 | m |
| **Solar field** | 256x360 | m |
| **Abscise Cloud initial location** | 0 | m |
| **North Cloud initial location** | 0 | m |
| **Irradiance without cloud shadowing** | 800 | W/m² |
| **Irradiance with cloud shadowing** | 300 | W/m² |
| **Ambient Temperature** | -3,75 | °C |

*Table 3 Environmental conditions*

At the beginning of the simulation it is considered a clear sky condition. However, the cloud in the sky is located next to the solar field ready to cross its solar surface. The cloud dimensions are considered as a square of 150 meters by each side. It is notorious that the cloud has a surface occupation of 24,4 % of the solar field surface, then it is expected to

have the entire cloud in the solar field during most time of the simulation.

As seen in Table 3, the cloud will cross in a diagonal path the solar field at a constant velocity speed. To be accurate the cloud moves from the south west to the north east of the power plant.

To finalize the weather conditions the simulation considers a solar irradiation of 800 W/m² when the cloud does not shadow the modules and 300 W/m² when the modules are shadowed, the ambient temperature is considered as -3.75 °C.

This environmental case could be related with a winter solstice in a quite northern or southern location. Obviously, it is considering the midday hour sun day. Otherwise, these values of irradiance would be really complicated to be reached.

### 7.1.1 Solving process

First of all, is simulated the cloud dynamics in order to know what will be the effect of the irradiance over the solar field.

After that, it is simulated the simplified PV generation model. This model considers a simplification of the solar modules. To do so, all strings have been simplified as described in 4.1.3 Simplified string . Then, each string considers as a solar module an amount of 30 modules and 30 times its power. To carry out this simulation the mean value of the irradiance for the whole string has calculated, as well.

After that, there exist sub variant of the previous simulated model where it can be evaluated any desired string at full detail. This simulation reflects the power generation in the chosen string allowing the fully detail irradiance input without being evaluated as a mean value.

Finally, the PV power generated in the solar field is driven to the VSC where it is transformed into AC low voltage power. The overall simulation is performed for 25 seconds.

## 7.2 Equipment

The equipment needed to simulate the problem exposed in the previous section refers only to that devices dedicated to the power generation or power conversion. The other equipment has been dismissed because it is not considered as fundamental equipment that can affect the final result and so the conclusions.

Therefore, the dismissed equipment is the electric wiring, combiner boxes, protection devices of the power plant, monitoring devices to control the production, fuses and the

trigging connection of the distributor.

### 7.2.1 PV module

The chosen module corresponds to the manufacturer Canadian Solar. The model is the CS3U-330P which has the following characteristics:

| Electrical Data STC | CS3U-330P | Units |
|---|---|---|
| **$P_{max}$** | 330 | $W_p$ |
| **$V_{mp}$** | 37,2 | V |
| **$I_{mp}$** | 8,88 | A |
| **$V_{OC}$** | 45,6 | V |
| **$I_{SC}$** | 9,45 | A |
| **Module efficiency** | 16,97 | % |
| **Operating temperature** | -40°C~+85°C | °C |
| **Max. system voltage** | 1500 | V |
| **Max. Series Fuse Rating** | 15 | A |
| **Dimensions** | 1960x992x40 | mm |
| **J-Box** | IP67, 3 diodes | - |

| Electrical Data NMOT | CS3U-330P | Units |
|---|---|---|
| **$P_{max}$** | 242 | $W_p$ |
| **$V_{mp}$** | 34,2 | V |
| **$I_{mp}$** | 7,08 | A |
| **$V_{OC}$** | 42,5 | V |
| **$I_{SC}$** | 7,63 | A |

| Temperature Characteristics | CS3U-330P | Units |
|---|---|---|
| **Temperature Coefficient ($P_{max}$)** | -0,41 | %/°C |
| **Temperature Coefficient ($P_{max}$)** | -0,31 | %/°C |
| **Temperature Coefficient ($P_{max}$)** | 0,053 | %/°C |
| **Nominal Module Operating Temperature (NMOT)** | 43±2 | °C |

| Assumed Characteristics | CS3U-330P | Units |
|---|---|---|
| **n** | 1,025 | - |
| **$R_{sh}$** | 200 | Ω |
| **$R_s$** | 0,26 | Ω |

*Table 4 CS3U-330P of Canadian Solar* [15]

This module also counts with 60 cells per each module. This fact is really important in terms of internal connectivity. It is assumed that as there are 3 by-pass diodes each module has

been split in three different parts that are formed by 20 cells combined with a by-pass diode, as shown in Figure 18.

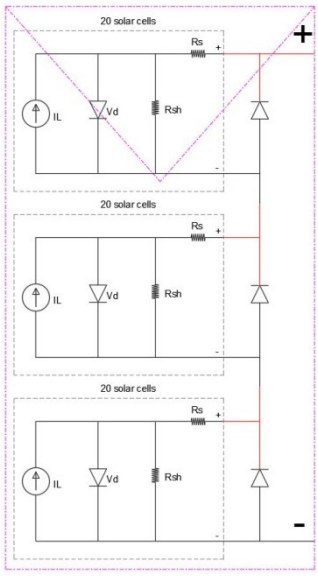


*Figure 18 CS3U-330P internal configuration*

### 7.2.2 PV Inverter

The chosen inverter corresponds to the General Electric manufacturer. Its electric device model is the LV5+1560.

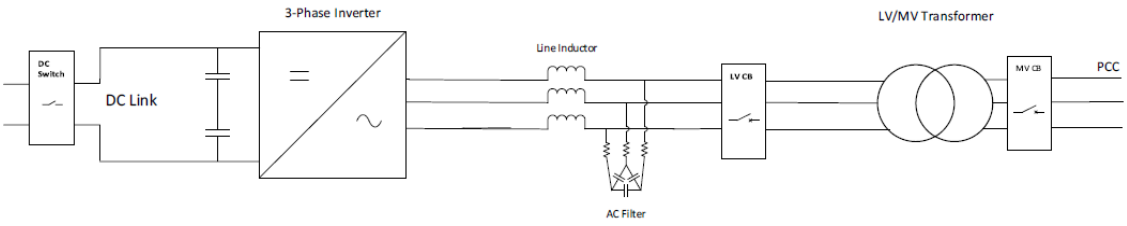


*Figure 19 LV5+1560 Inverter simplified modeling*

The general scheme of the inverter has been simplified by discarding the AC filters, the circuit breakers and the power transformer from low voltage to medium voltage.

| Input Data | LV5+1560 | Unit |
|---|---|---|
| **MPPT Range** | 880-1300 | $V_{DC}$ |
| **Max Permissible DC Voltage** | 1500 | $V_{DC}$ |
| **Max Continuous DC Current (at 35°C / 50°C)** | 4000/3200 | $A_{DC}$ |
| **Max DC Short Circuit Interrupt Rating** | 12000 | $A_{DC}$ |
| **Number of MPPT** | 1 | |
| **Number of DC Inputs** | up to 24 | |

| Output Data - Low Voltage | LV5+1560 | Unit |
|---|---|---|
| **Active AC Output Power (PF=1) (at 35°C / 50°C)** | 3,12/2,76 | MW |
| **AC Output Voltage (+10% / -10%)** | 600 | $V_{AC}$ |
| **Max AC Current (at 35°C / 50°C)** | 3000/2655 | $A_{AC}$ |
| **Grid Frequency ±5%** | 50 | Hz |
| **Power Factor (PF) Range** | 0-1 | |
| **Current Harmonic Distortion (TDD)** | <3 | % |

| Protection Rating and Ambient Conditions | LV5+1560 | Unit |
|---|---|---|
| **Operating Temperature Range** | -25 to 50 | °C |
| **Storage Temperature Range** | -40 to 65 | °C |
| **Cold Weather Option** | -35 to 50 | °C |
| **Humidity** | 5 to 100 | % |
| **Maximum Altitude Without Derating** | 2000/6562 | m/ft |
| **Seismic** | Zone 2B ASCE 7 / IBC | |
| **Maximum Wind Speed** | 250/155 | kph/mph |
| **Snow Load** | ASCE 7 | |
| **NEMA Rating / IP Class** | NEMA 3/IP54 | |

| Efficiency & Auxiliary Power | LV5+1560 | Units |
|---|---|---|
| **Inverter Efficiency (Max / EU / CEC)** | 98,9 / 98,6 / 98,7 | % |
| **Nighttime Aux Power** | <200 | W |

*Table 5 LV5+1560 datasheet* [16]

Regarding that the general information is given in the main datasheet of the product. Some input data not known in Figure 19 has to be assumed in order to carry on the simulation. The unknown data is that data about the passive elements of the circuit such as inductors, capacitors…

Besides, the controllers for the inverter have also been assumed due to the fact that no information is given by the manufacturer. The taken values are approximated to other facility features of similar power conversion.

| Assumed input data | LV5+1560 | Units | Assumed input data | LV5+1560 |
|---|---|---|---|---|
| **Line inductor** | 36 | µH | **$Kp_{PLL}$** | 1 |
| **Capacitor DC link** | 50 | mF | **$Ki_{PLL}$** | 2 |
| **Line resistor** | 0 | Ω | **$Kp_{DC}$** | 5 |
| **Inverter volatge** | 612,8<0° | V | **$Ki_{DC}$** | 10 |
| **Inverter peak voltage** | 866,63 | V | **$Kp_{CCL}$** | 0,012 |
| | | | **$Ki_{CCL}$** | 0 |

*Table 6 LV5+1560 Assumed data*

### 7.2.3 Structure

The structure does not affect in a direct way the power production by means of electrical conversion but it performs the technique to take advantage of the incoming irradiance.

In this case it is been assumed a tracker for the layout configuration. These trackers have a pitch of 5 meters and a total length of 360 meters, as exposed in 7.1 Problem to be simulated.

In Figure 20 it is observed that the magenta lines represent the tracker layout on a rectangle surface of 256x360 meters. Whereas white gaps are the corridors left to make the maintenance and the correct tracking orientation.

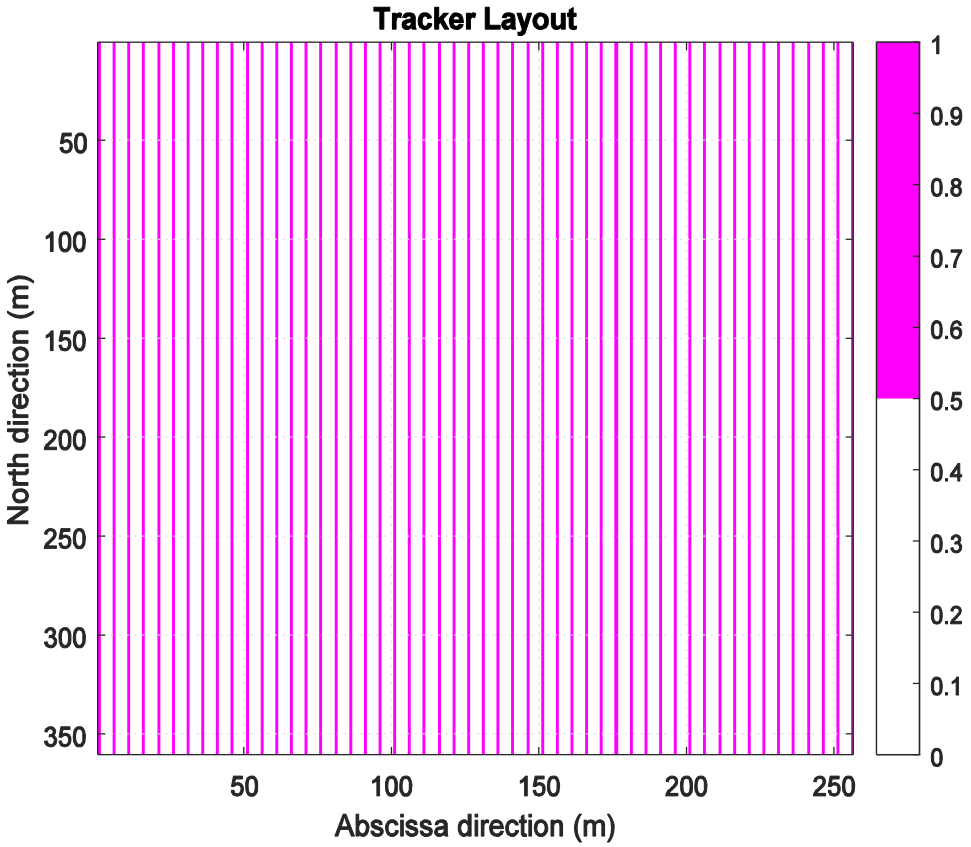


*Figure 20 Tracker Layout*

## 7.3 Simulink diagrams

This section shows the main Simulink diagrams that are performed for the modeling of this thesis. Many of them are presented in a general way because of its huge extension on the Simulink screen block diagram.

It is important to remark that these simulations are joined by their output obtained data. This fact means that the topics presented are evaluated and the generated information is extracted from the model. Before beginning the following process the output data that was downloaded from the previous simulation is uploaded in the desired simulation to be carried out.

### 7.3.1 Simplified PV block model

The simplified PV block model represents the production of the whole power plant. This model is the most important model for several reasons.

Firstly, it contains the whole strings of the whole power plant. All these strings are connected in parallel to the same MPPT because the inverter to be evaluated has one MPPT.

As shown in Figure 21, the string model configuration is modeled like one general solar module as explained in 4.1.3 Simplified string configuration. Another model block has been attached to the module on order to measure the current, voltage, energy and power outputs.

The most important values obtained are the power and the current which are related each other. The voltage measurement is not mandatory at all because it is imposed by the MPPT algorithm but one can compare both voltages and realize that they are equal.

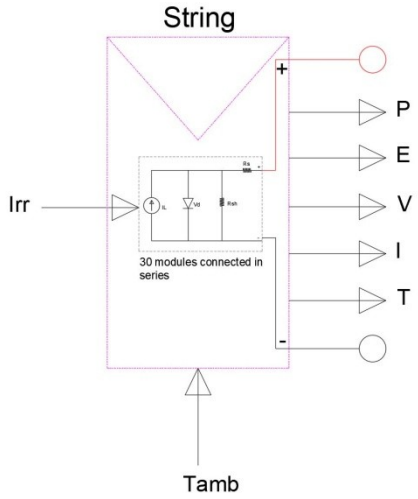


*Figure 21 String simplification of 30 modules*

Secondly, the model associates the input irradiance to every module location which means that every module has been located at a given coordinates. By means of the input solar field irradiation matrix the irradiance is applied at a certain module.

As it can be observed in Figure 22 every string has a different input irradiance variable. This input variable depends on each module location and its mean irradiance combination so as not to generate false output values.

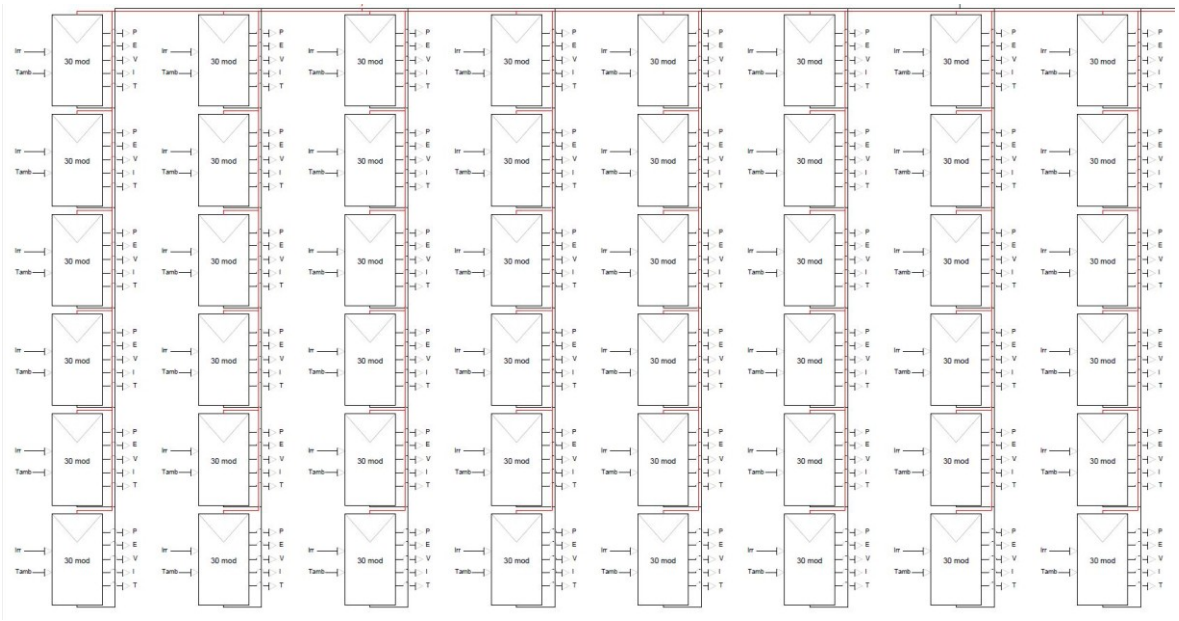

*Figure 22 First rows of trackers*

Figure 22 shows the first strings of the power plant. All the strings that are in the middle to the end are not shown because as it can be observed the strings are forming a pattern of six strings in the vertical axis as designed in the tracker layout, Figure 20.

Thirdly, the model associates every input irradiance value to an output power production. The importance of the model relies on the amount of information that is working with and the amount of information that is being generated. It is really important to have it really organized. Otherwise, the outputs could be really confusion when trying to analyze what is going on.

Figure 23 shows how the output variables for every column tracker are gathered organized in just one variable to be treated. The first gathering is produced at the first column of trackers, then the second gathering is the corresponding with Figure 23.

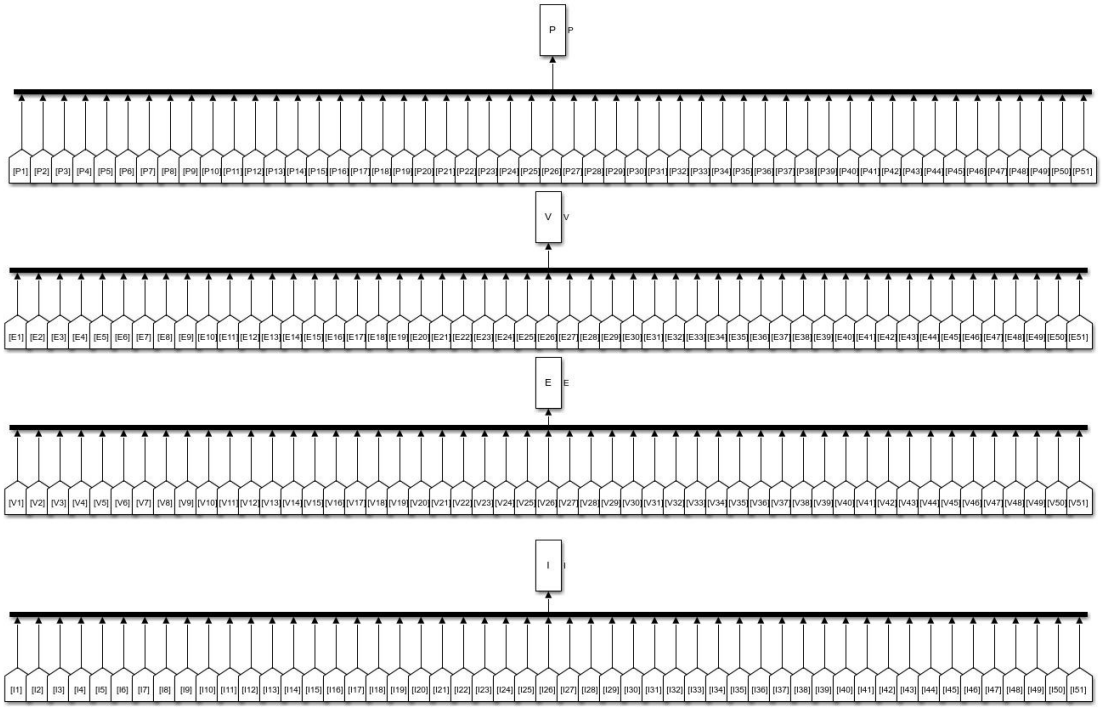


*Figure 23 Output data of the model*

Finally, this model processes the whole power generation that will provide substantial results of what facts are happening on the power plant.

It has to be remarked that as the simulations have been performed separately the dynamics in the DC voltage is lost because the delay time is not involved in the DC voltage.

### 7.3.2 Detailed PV block model

The detailed PV model is a variant of the previous model in order to focus on an accurate precision about the behavior of the string. This model is addressed to study a specific string, so it has to be chosen the string that it is desired to study.

All strings can be addressed but it is understood this model to focus on the one that could give some kind of trouble or it could be a specific interest.

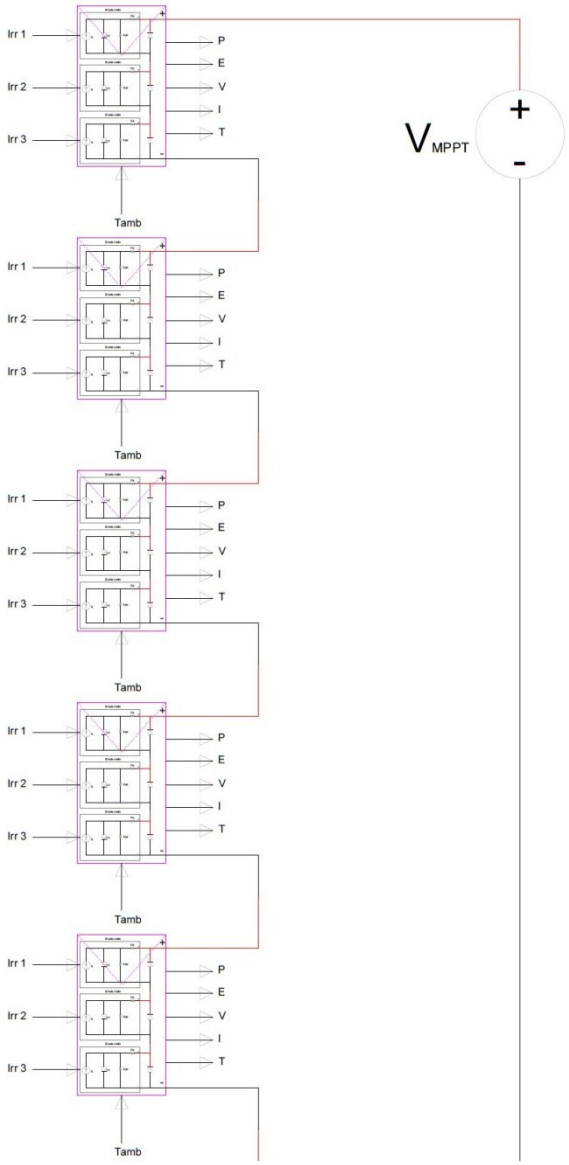


*Figure 24 First modules of the detailed string*

Apart from choosing the desired string, the model has a similar functioning compared with the previous model.

Nevertheless, it has to be known that it is needed to simulate the previous model in order to simulate the current one. The reason is because this model does not include the MPPT algorithm which is fed with the voltage of the previous model. This fact allows simulating the same string in the same voltage conditions while the irradiance variable is maintained from the solar field matrix. The outputs of the 7.3.1 Simplified PV block model works as inputs in this model.

This operating mode gives a better resolution of the current output and power production of

each module that in this case is not modeled as it is been shown in Figure 21. In the current simulation the module has been readapted to what it is been explained in 4.1.2 Manufacturer module configuration, also shown in Figure 26.

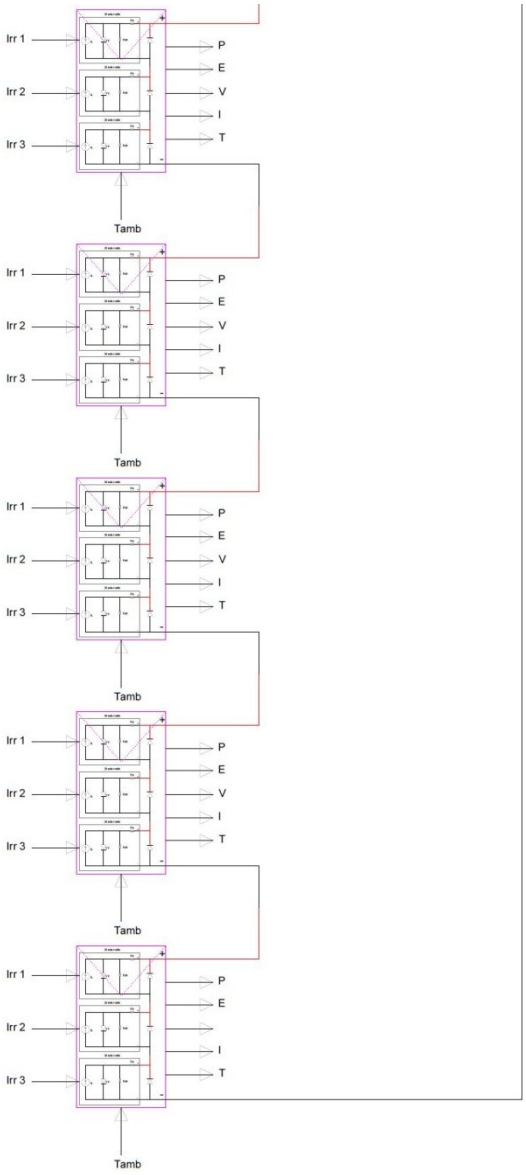


*Figure 25 Last modules of the detailed string*

The model consists on the grouping of 30 modules in series from the first module shown in Figure 24 to the last module shown in Figure 25.

Additionally, it can be observed that this model includes three input irradiance variables per module. As said before, the input irradiation matrix is maintained but the way this matrix is treated has changes substantially.

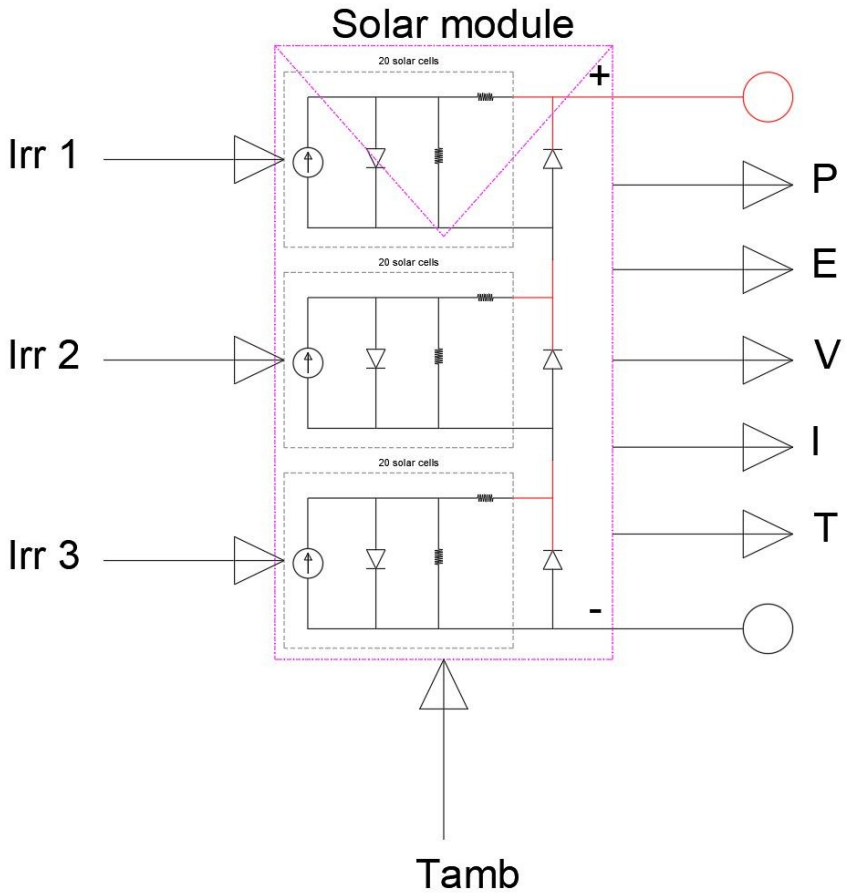


*Figure 26 Detailed module with measurements*

In this case the irradiance is not taken as a mean value. The irradiance is been calculated accurately and, then introduced in the specific part of the solar module. Although there is one module, the amount of irradiation variables that are introduced in each module depends on the by-pass diodes. This fact produces a better precision on the power production simulation and the current output of every part of the solar cell.

Another important part of the model is that it analyzes the power generation of each module instead of a full string. If it is compared module models one can observe that the simplified simulation measures one cell part and in this simulation three cell parts are measured. The reason of not doing one per each part is because the module is understood as one element but it could be measured if it was desired.

Finally, all proceed data is been organized stored for the same reason that was organized in the previous chapter. In Figure 27, it is observed that every module is organized in a vertical direction to create a unique variable that will be able to be treated in case of being desired without the necessity of many variables.

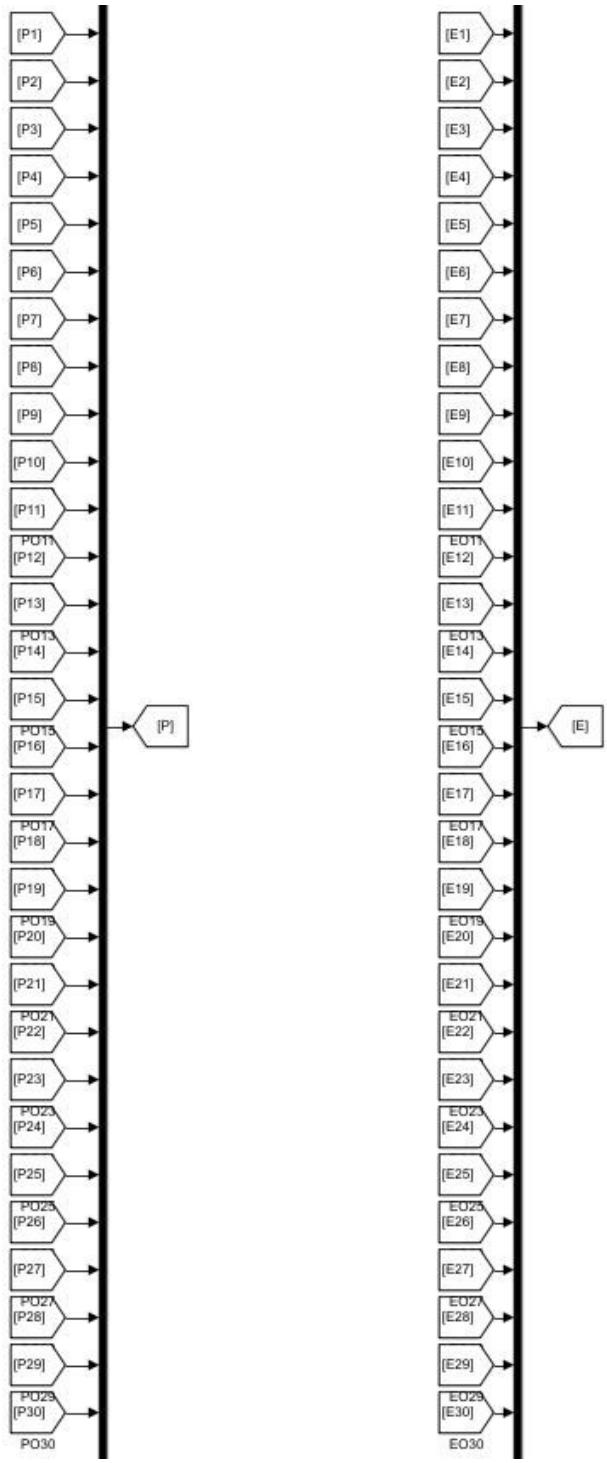


*Figure 27 Output power generation of the detailed string*

### 7.3.3 VSC block model

The VSC model consists of three main parts in the same model that permits the control of the inverter and the conversion of the DC power to AC power.

The inputs in this model that are coming from the simplified model are the MPPT voltage and current generated. These are the only inputs needed from the PV modules DC side of the power plant, so other processed data such as the irradiation in the solar field coming from the beginning is not necessary anymore.

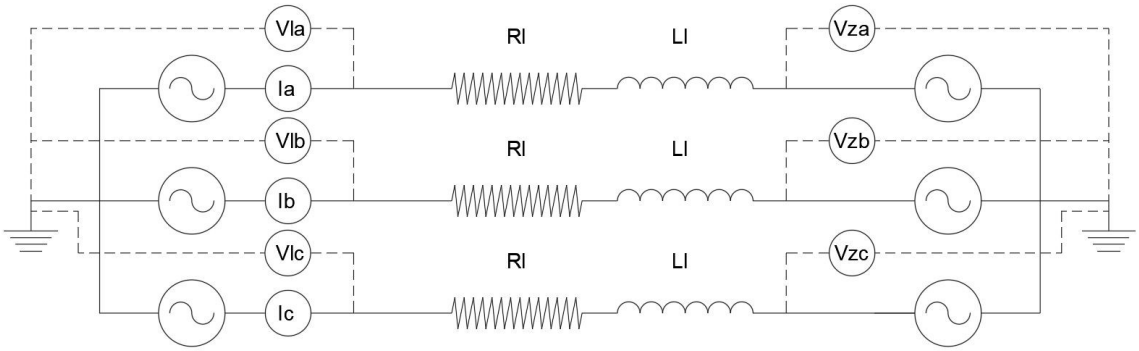


*Figure 28 VSC connected to the network*

The first part of the VSC model is referred the connection between the grid and the AC side of the inverter. This evaluates the voltages measured from the grid and the voltages generated by the AC side of the inverter.

The second part is referred to evaluate the computed current of the VSC. As shown in Figure 15, the reference current of the inverter is directly connected with the current and voltage generated in the DC side of the power plant.

The last part is referred to the inverter control part being the most complex among the three of them.

Compared with the other models this model does not need the usage of low-pass filters to work with data of the previous time domain. As this part does not include the same algorithm loops that the other models contain it can be simulated by using memory blocks.

The control algorithm to be simulated follows the same scheme as shown in Figure 12, which is related to the third part of the model. All this control is implemented with the content exposed in 5 Grid-connected voltage source.

This model does not work with the same amount of information and variables that the other models worked with. Then, the generated data is more manageable and not as difficult to work with.

## 7.4 Simulated results

The results and presented graphics are exposed for each simulation part. Furthermore, they are explained in a coherent developed methodology allowing understanding what has to be analyzed before starting the following process.

As this simulation is a dynamic simulation that requires for graphic variations in many cases some examples at time sequence of 14 seconds and 20 seconds are provided. However, the whole simulation consists on animated plots that are constantly changing throughout the simulation.

### 7.4.1 Cloud dynamics

By having described that the cloud is moving in a straight line, Figure 29 shows the real path that is mapped upon the tracker layout. It is observed that the cloud affects one part of the modules in their whole followed path.

The path affects in a direct way all modules located under its shape. However, the cloud affects the overall power plant that is described in 7.4.2 Simplified PV power plant, as well.

During the simulation the whole cloud is found into the solar field for several time because the cloud is crossing the tracker layout in a diagonal direction and its dimensions are smaller than the tracker layout dimensions.

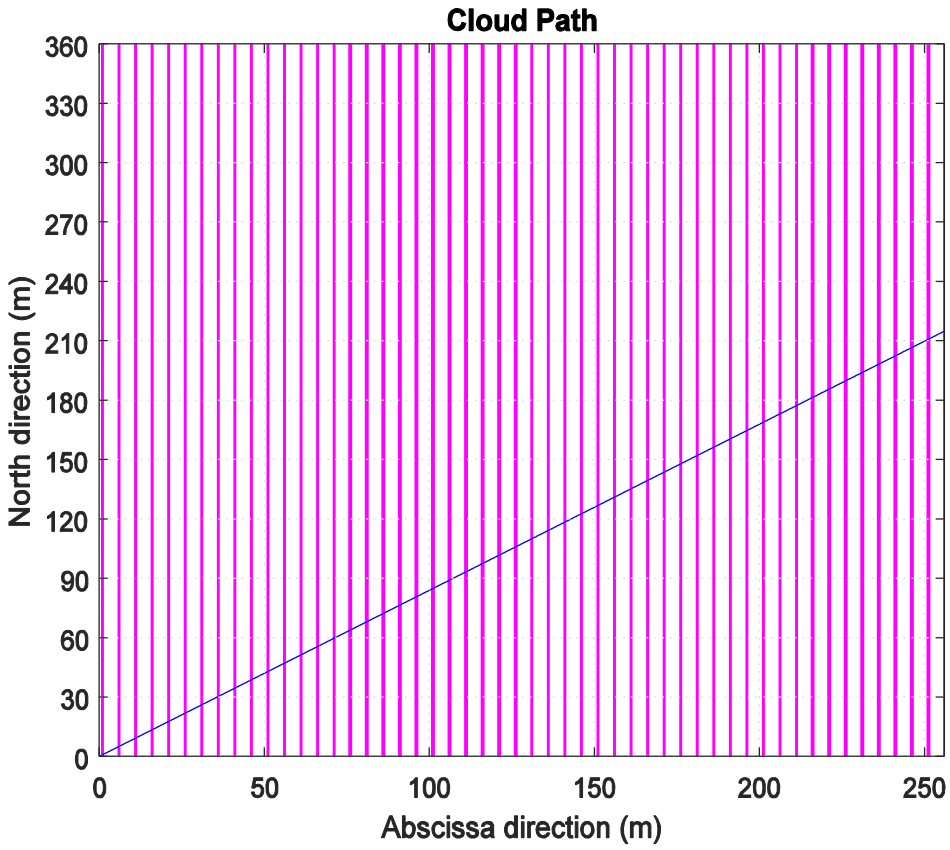


*Figure 29 Cloud path*

In Figure 30 it is observed the dimensions of the cloud upon the tracker field. In this case, the whole cloud is located into the surface of the tracker field. Besides, it can be easily compared that this cloud is not a really big cloud, so its effects will be evident but will not be catastrophic from the power production point of view.

The cloud has the potential to affect not only the direct modules that are shadowed but the strings that are connected to a shadowed module. Therefore, the string connection is also an issue when designing the module layout disposition.

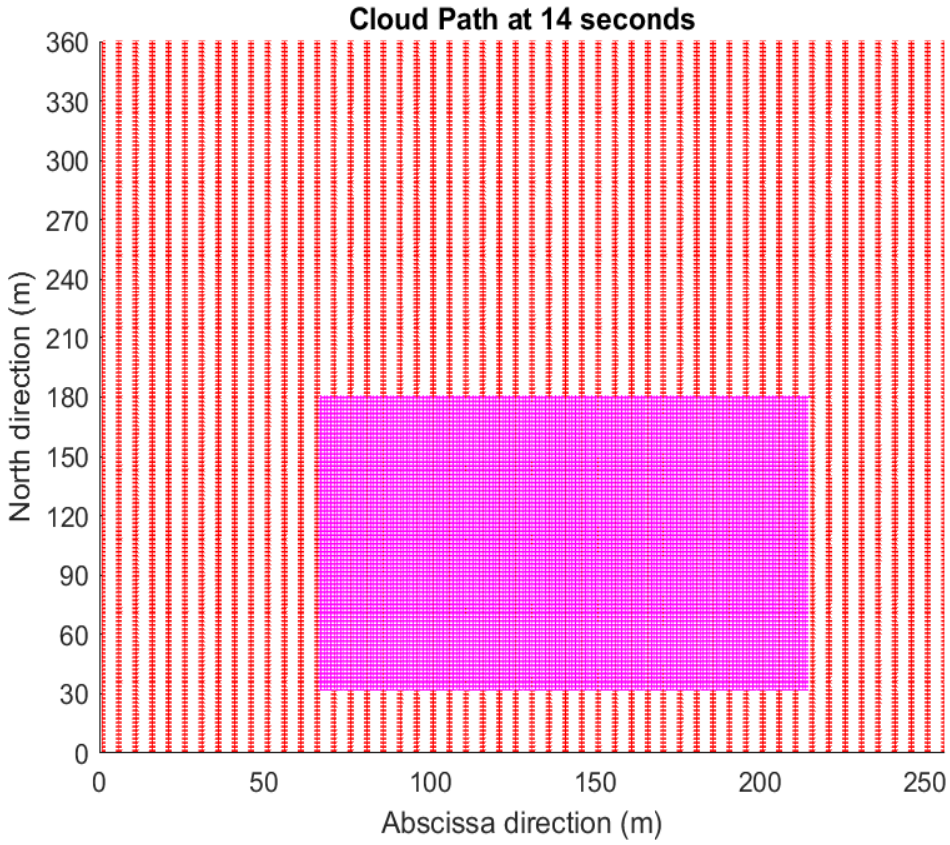


*Figure 30 Cloud shadow on the tracker field*

When the irradiance coming from the sun is taken into account upon the solar field and the cloud makes the effect of trapping and scattering some incoming irradiation is it obtained an irradiation map like the one shown in Figure 31.

If it is compared Figure 30 and Figure 31, it is clearly seen how the cloud matches with the irradiance drop in the solar field while the area not shadowed remains at the incoming irradiance.

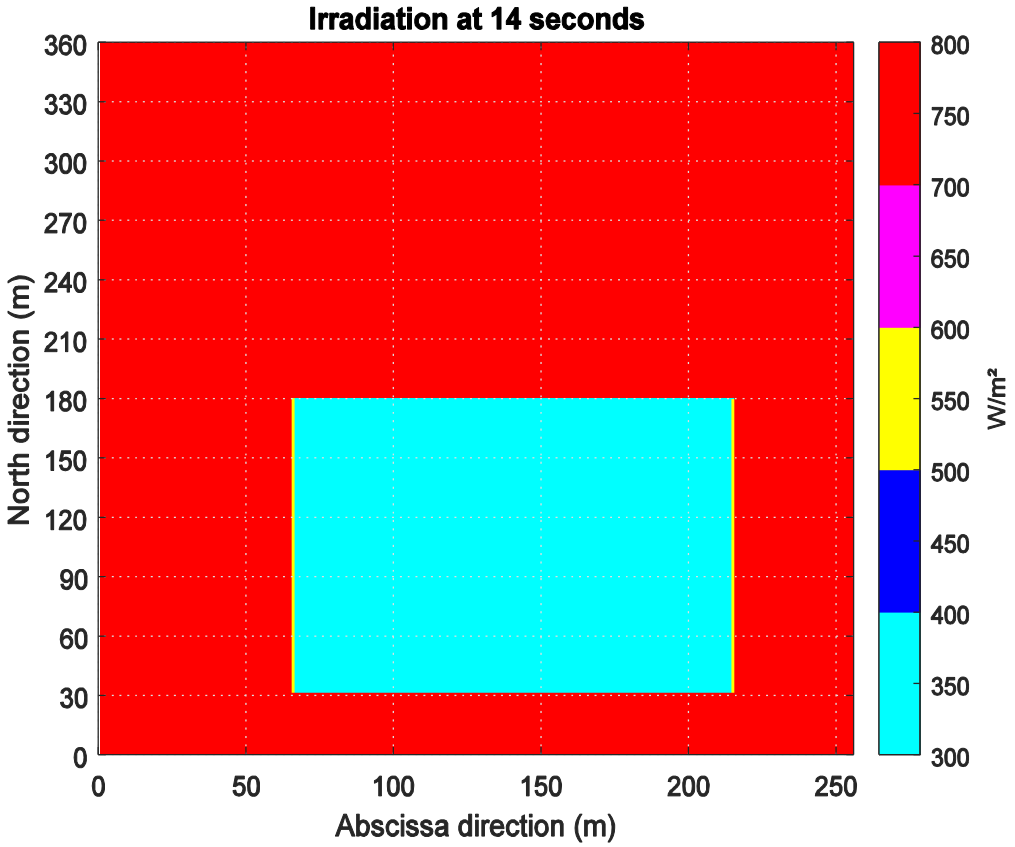


*Figure 31 Irradiation on the solar field*

When this irradiation matrix on the solar power plant is matched with the module coordinates it is obtained that some irradiation in the irradiation matrix does not contribute in the power production.

This non contribution allows the PV power plant not to be in a constant changing power situation due to the trackers gap.

As shown in Figure 32, the movement of the cloud does not affect as continuously as it affects the solar irradiation in the whole solar field.

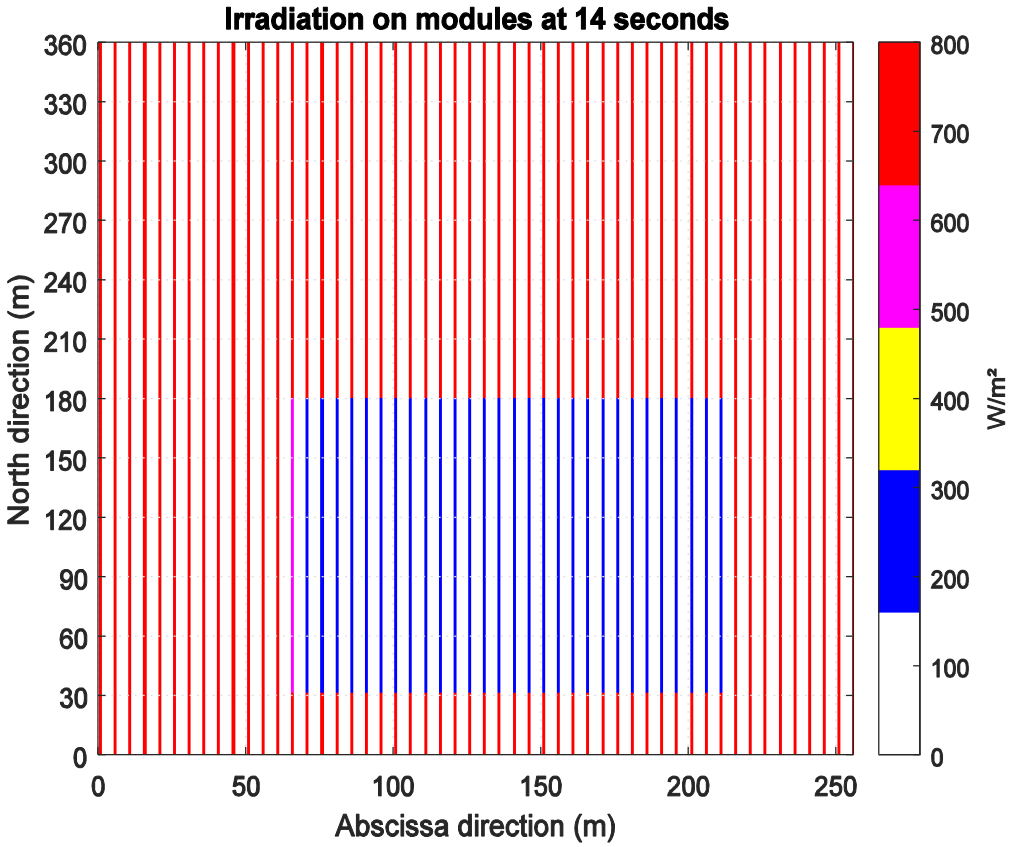


*Figure 32 Irradiation on the solar modules*

The irradiance projected upon the solar modules is the irradiance linked with the simplified model PV to be simulated. These irradiances are reference coordinated with each module location in order to accurately simulate the main PV simulation.

Another analyzed parameter is the module temperature for each module in the entire power plant. Temperature affects the power generation depending on the ambient temperature and the incoming irradiance. The lower the temperature the higher the DC power extraction. On the other hand, the higher the temperature the lower the DC power extraction.

As shown in Figure 33, the temperature difference is produced in those modules of that are receiving a lower irradiance. It is observed that the module temperature at 800 W/m² is 19 °C while the temperature when the irradiance decrease to 300 W/m² is about 7 ℃.

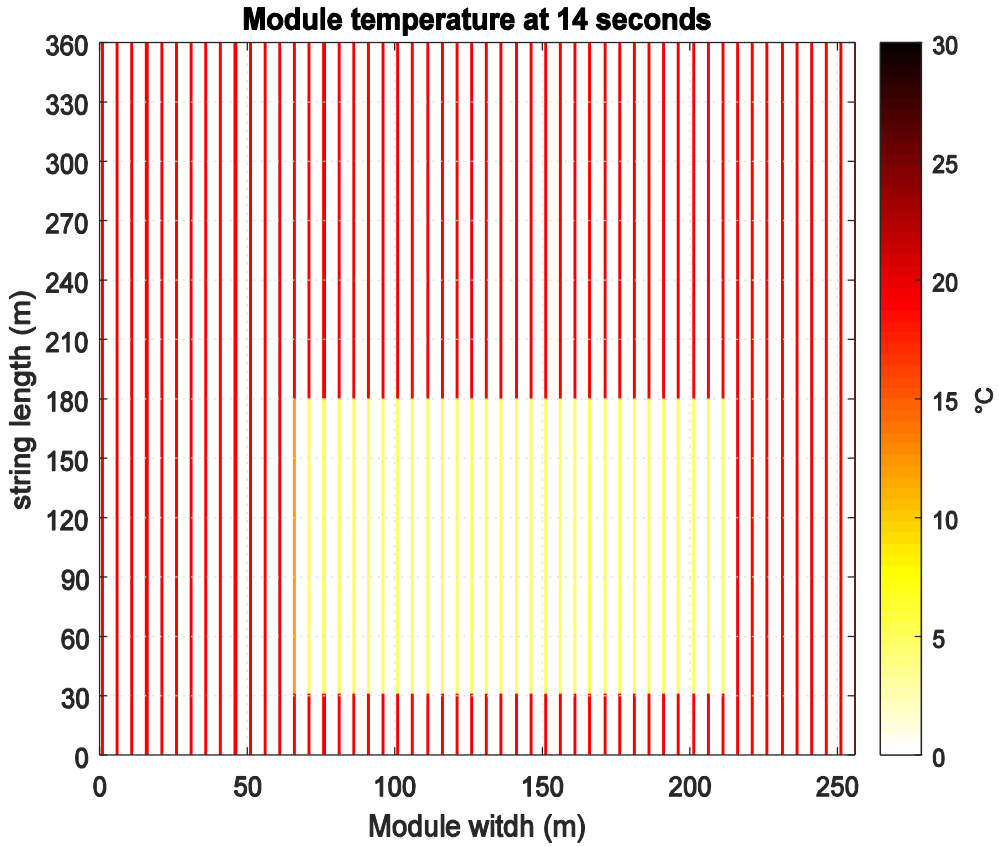


*Figure 33 Module temperature in the overall power plant*

The most important fact about the temperature difference is that these incremental differences in the solar cell are poisonous for the modules because of the hot spot effect [17].

This particular case is not considered as problematic because the temperature difference is not really high and its shadow lasts a small instant. The pessimistic case would be more problematic at higher temperatures differences and more during time shadows because of the internal dilatations.

### 7.4.2 Simplified PV power plant

This simulation evaluates the power generation by means of the MPPT algorithm. Before doing so, it mixes the incoming irradiance from the previous simulation with the necessary PV equipment.

After that, the simulation starts the processing and at the very beginning the MPPT starts its functioning.

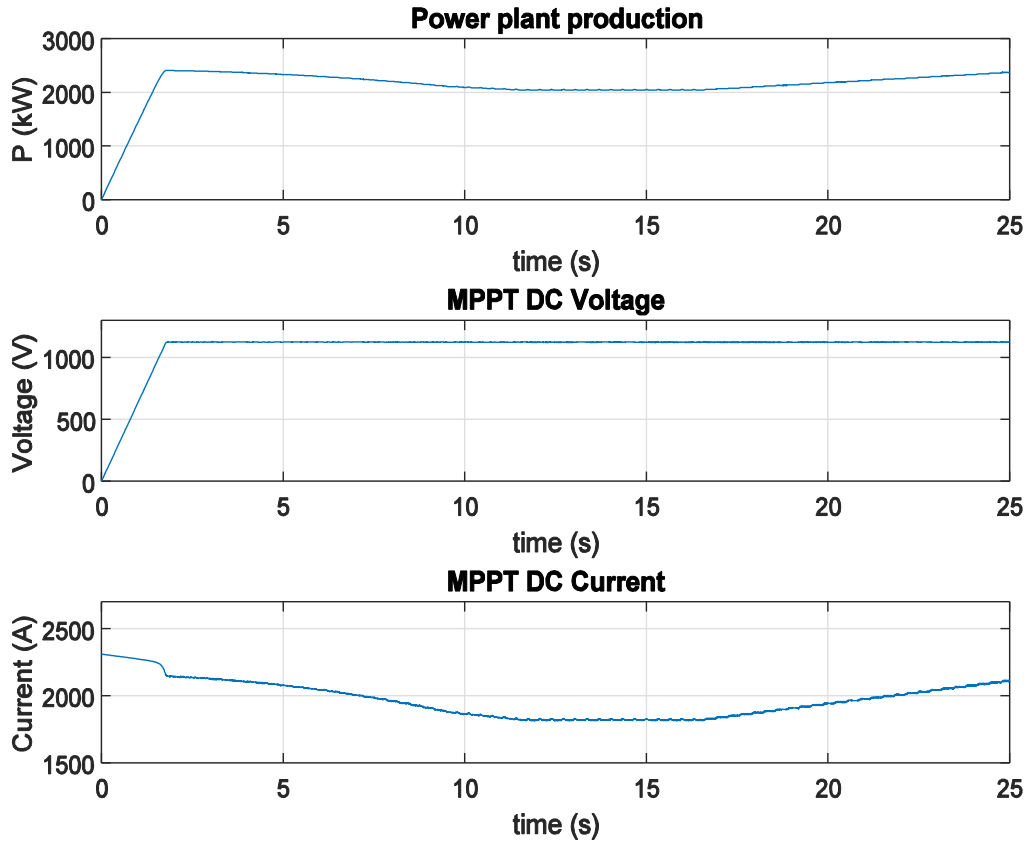


*Figure 34 Power plant P-V-I MPPT measurements*

From Figure 34, many issues are observed from the starting time when there is no presence of the cloud until the end of the simulation where the cloud is abandoning the solar field.

Approximately, from time 0 seconds to time 2 seconds it is seen how the MPPT algorithm starts from 0V to search the maximum power point. The reason it takes 2 seconds is because of a programmed issue, it could be reached with lesser time but it was desired to show how the algorithm also incorporates a process that evaluates the tracking algorithm for different voltage inputs compared with the voltage at the maximum power point.

Although, the cloud is entering in the solar field at the same time is doing the MPPT its entrance is not sensitive because of the MPPT algorithm is not in its maximum power point yet.

From second 2 to almost 12 it is observed how the MPPT is reached and the cloud is still entering in the solar field. An interesting fact in this commented timeline is that the voltage

remains constant all the time while the variation is produced at the generated current.

From second 12 to 17 it is not seen any power variation which means that the whole cloud is inside the solar field and it is moving inside. Therefore, the generated power cannot be additionally decreased.

From second 17 to 25 the cloud starts to go out from the solar field and power gains are rapidly observed.

The general result observed it that the effect of the cloud during 25 seconds has a fast response on the generated DC power. The power variation produced is about 200 kW of power, in case the cloud was much bigger the effect observed would have been much higher and it would be observed a variation of MWs. Furthermore, these effects are directly reflected to the network which has to adapt to such intense power variation for environmental causes not controllable for humans.

During the whole simulation it is observed that voltage remains constant and the variations are produced on the current measurements. Knowing that shadows can affect these measurements it is been performed a variant simulation that shows the V-I curve and the power production for every voltage value.

Figure 35 shows that the traced curve is not like the typical V-I curve given by the manufacturer. Although, it takes into account all modules in the power plant it also considers the cloud effect. Hence, as seen from voltage 0 V to almost 100 V there is current drop on produced by the cloud effect. In case, the existing cloud would be much bigger the current drop would be much bigger, as well.

It also has to be said that this Figure 35 depends on the studied conditions. So, if there was a displacement of the cloud, a variation in the incoming irradiance or any other environmental variation, the graph would change in a minor or major way.

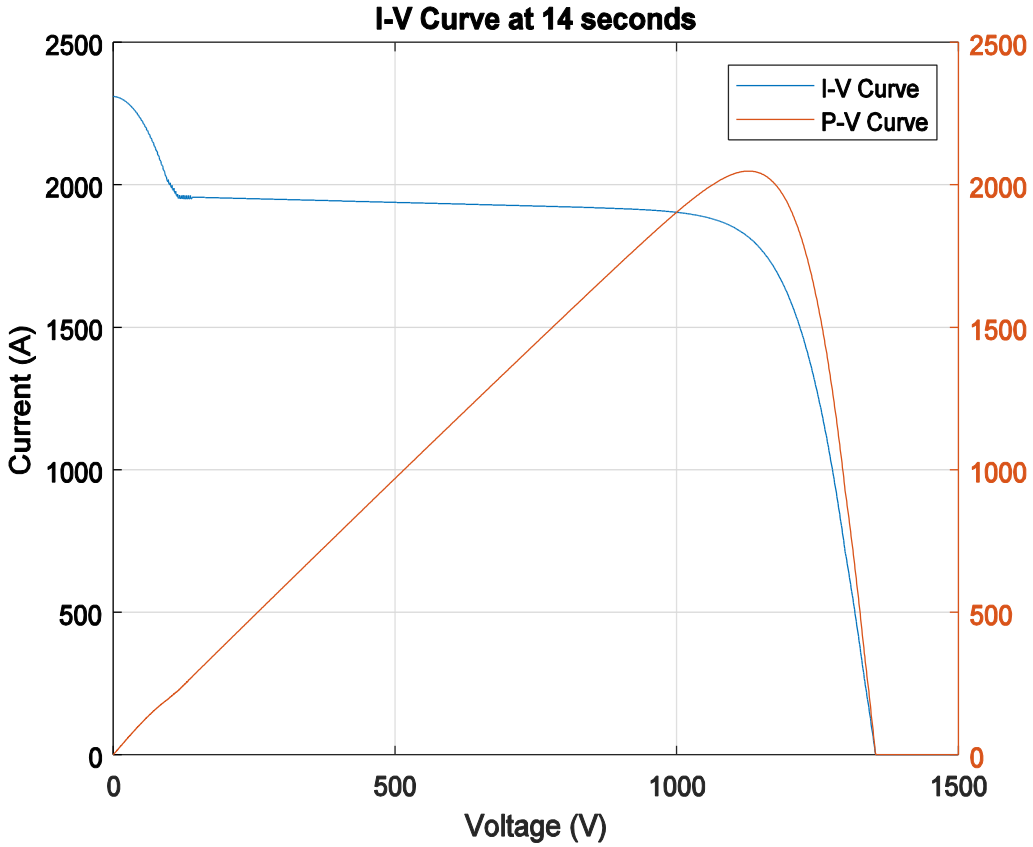


*Figure 35 P-V-I curve at 14 seconds*

The simplified simulated power plant also generates the power generation distributed among the tracker layout.

As all strings are connected in parallel, the main effect of the cloud is just produced under it and its surroundings. However, as shown in Figure 35 the ideal power generation would have been at a current of about 2300 A and 1200 V, approximately. In this particular case, the effect of the cloud produces an increase of voltage compared with the optimal solution without clouds. This small increase produces a small power drop in the other modules that are not shadowed in the power plant. This effect is considered an indirect effect of the cloud because it cannot be seen if it is not analyzed.

The optimal power production for every string is 7920 W but as it can be observed in Figure 36 the power produced for those strings not shadowed is 7770 W. This implies a reduction of 150 W per each module like similar to lose two thirds of a modules per each string in STC.

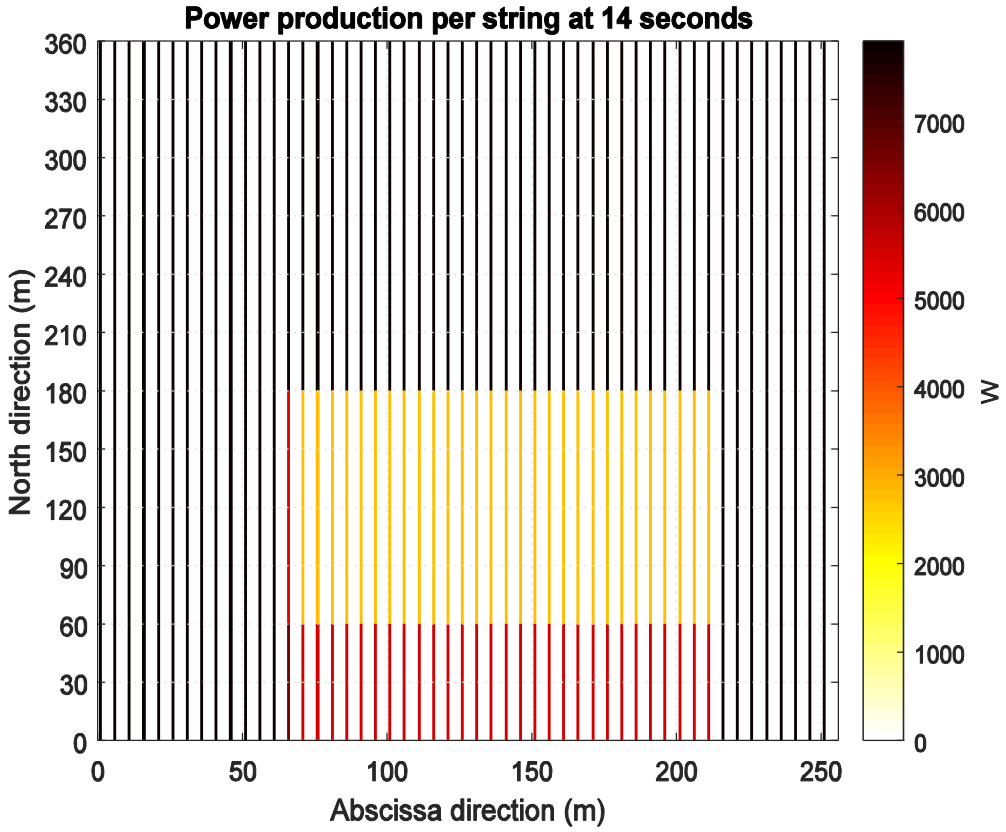


*Figure 36 Power distribution per string*

Moreover, it is easier seen the effect of the partial shading in some strings. Those strings in purple at the bottom part of the graphic show that the cloud is leaving those strings and, consequently the power produced is much bigger due to the increase of irradiance.

Finally, it is projected the overall energy produced per each string during the whole simulation in Figure 37. Obviously, the energy produced in the whole simulation is not quantitatively enormous but it is interesting to know its impact. The total amount of energy generated within the 25 seconds of simulation is 14.76 kWh. The mean paid price according to OMIE in the year 2019 is 47.68 €/MWh [18], which means that the amount money corresponds to 0.7 €.

The conclusion by seeing the generated money with the evaluated time is that in a short time the shadows will not have a huge impact because gains are not huge but in a time durable scenario it could affect money gains substantially.

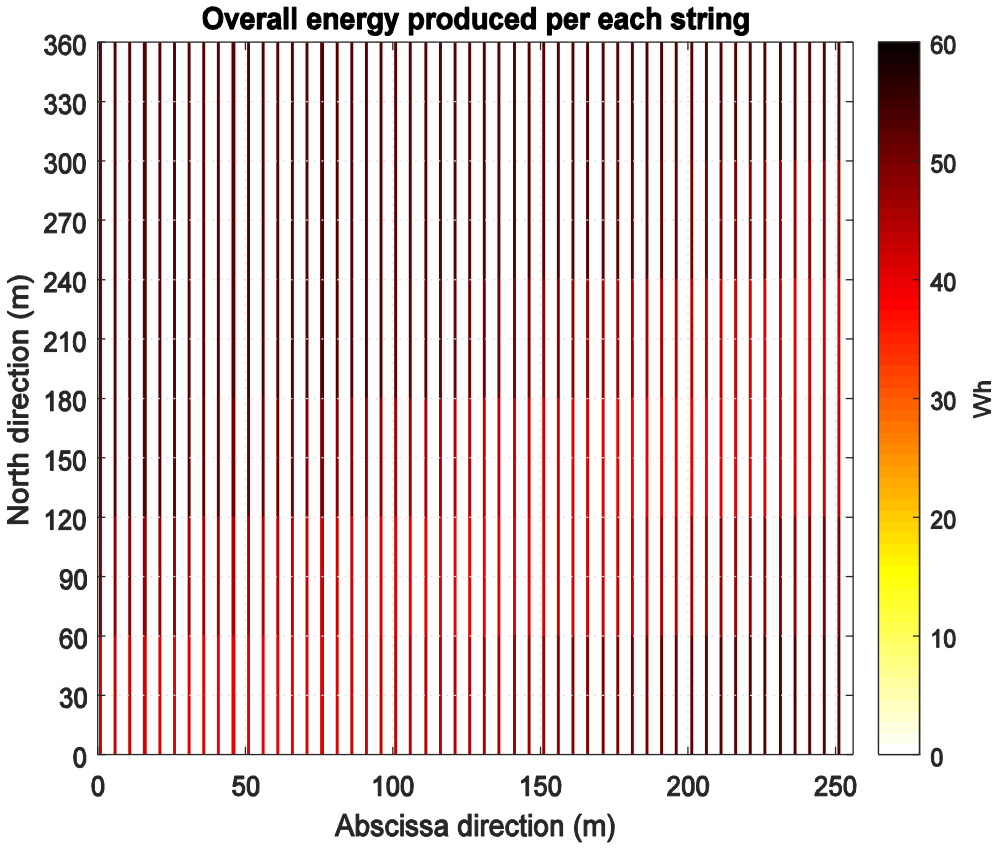


*Figure 37 Energy produced per each string*

Figure 37 also shows that the strings that are producing more energy are obviously these that were not affected by the cloud shadow. It is clearly seen the path of the shadow in the figure and its energy production decrease.

All these figures are taken from several animated video plots that were generated in Matlab and they can all be seen in the following Power Plant reproduction list. This reproduction list gathers those videos related to the dynamic changing behavior time domain in the entire power plant.

### 7.4.3 Detailed string simulation

This case is a more focused facing simulation to a desired string. Being understood that the previous simulation does not make deep focus on each module but on strings.

The analyzed string it the number 209 which corresponds to the string situated in the column number 35 and the 5$^{th}$ row. It is been chosen for the main reason that it is directly affected by the cloud shadowing.

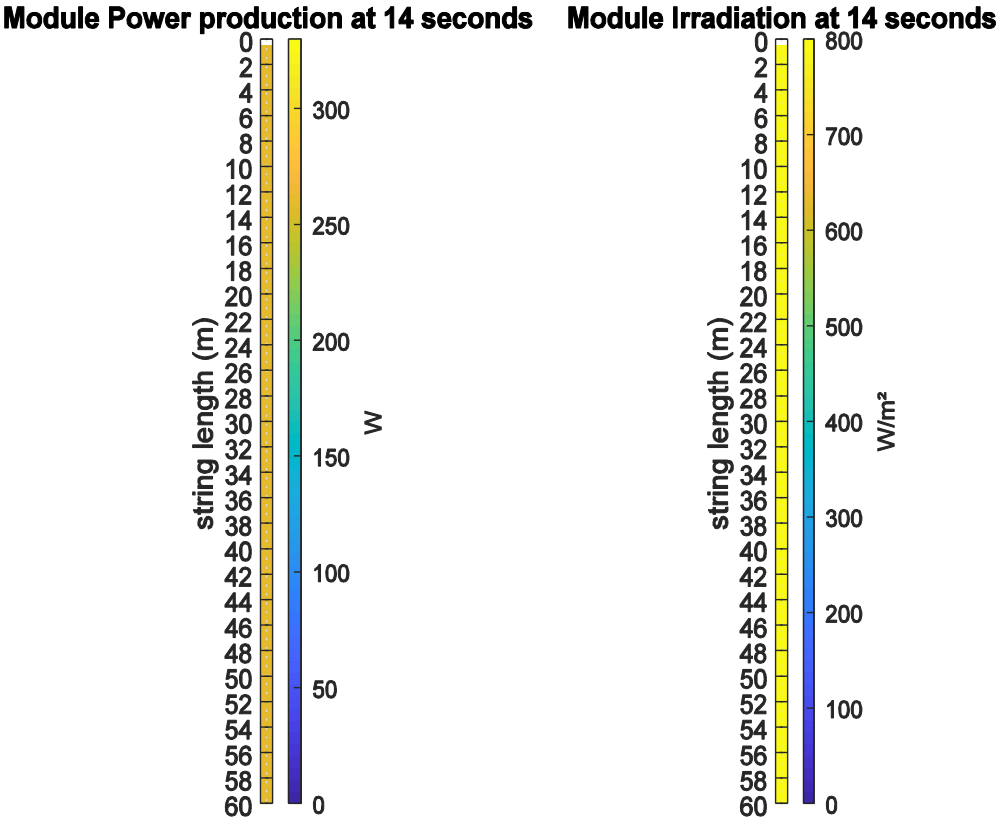


*Figure 38 Power and irradiance per module at time 14 seconds*

In Figure 38 it is shown how the 30 modules are still not shadowed on the right string block. Its incoming irradiance corresponds to the incoming irradiance coming from the sun. However, it can be observed that the generated power is 259 W while its maximum power in these environmental conditions without clouds is 264 W. The difference looks ridiculous but the power drop during large amounts of time represents important amounts of energy not being produced along a year.

As at this time the string is not partially or fully shadowed any additional effect is not seen in this part.

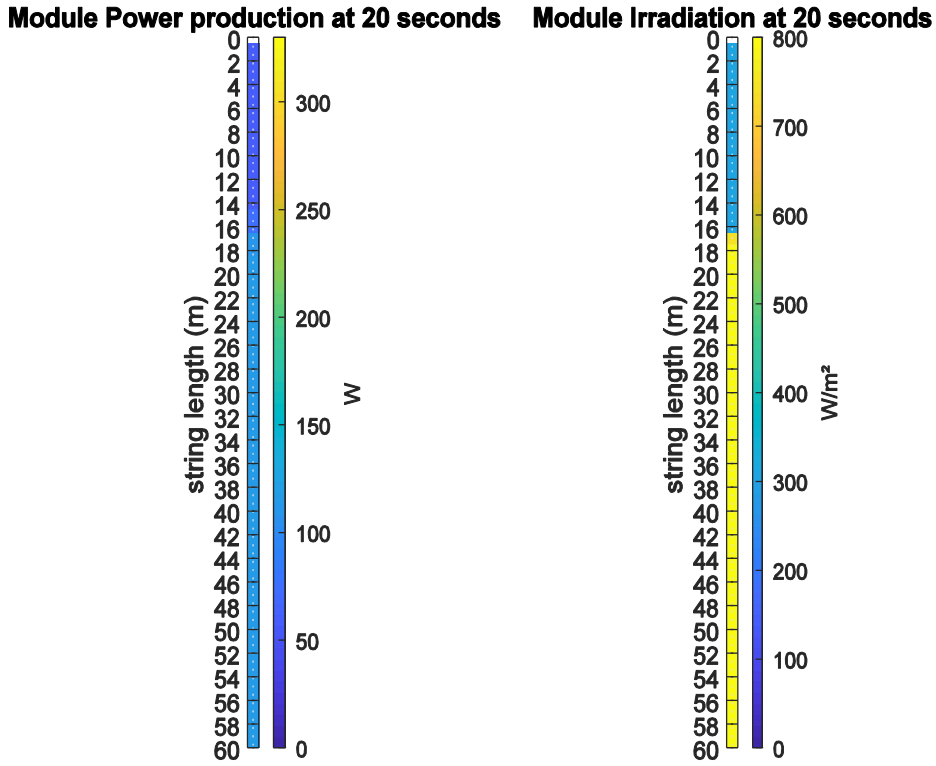


*Figure 39 Power and irradiance per module at time 20 seconds*

At time sequence 20 seconds it is observed how the power has changed from the time sequence 14 seconds. In this case, the cloud is partially shading the string. The first 8 modules are producing an amount of power 60 W each module and the rest are producing 116 W. Figure 39 shows the difference between of irradiance among the modules and as some modules are receiving 800 W/m² its output power is being affected by those modules that are shadowed.

The reason that the string that represents the power output is colored in blue is because the string color bar is compared with its maximum power at STC. That is why it is quite difficult to see the differences of power productions among modules.

As implemented in the overall power plant, in the string part is also seen the temperature effect from a higher accuracy in Figure 40.

It is notorious the temperature difference among modules at time second 20 where some modules are receiving less irradiance than others. As explained in the general case, its temperature difference is not problematic in this particular case but it has to be controlled

that hot spot is not an issue for these modules.

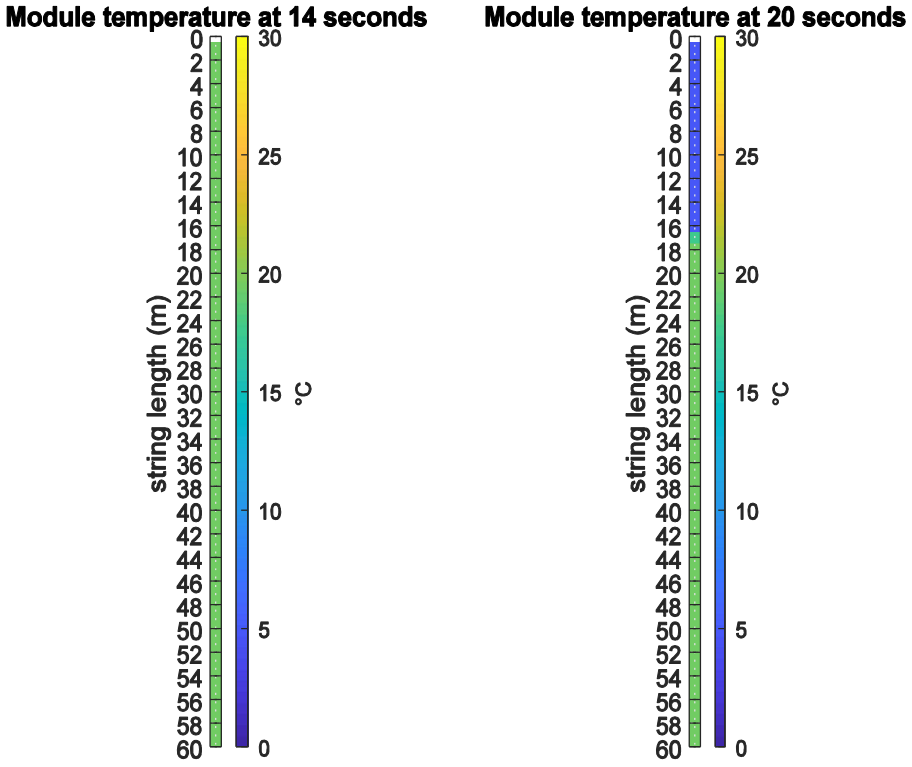


*Figure 40 Module temperature*

In terms of energy production in the chosen string, the overall energy produced is 49.37 Wh which is similar as gaining 0.0024 € in one string during the simulation time. After being analyzed the chosen string, it is observed that the cloud is not shadowing its entire surface for any specific moment, it is seen that the shadow effect has a minor effect and consequently variations in the module energy production are not even aware in Figure 41.

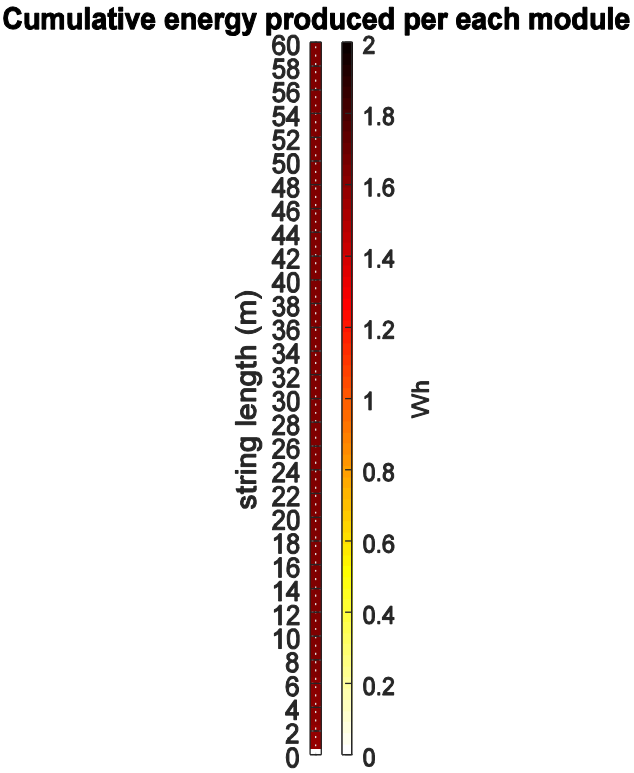


*Figure 41 Cumulative energy produced per each module*

Approximately each module is producing 1.65 Wh during the simulation. As seen in Figure 39, there exist an effect of the cloud in some modules but as this shadow effect last an instant because the cloud hardly passes over, the effect on the energy production is seen almost neglected in Figure 41.

Regarding the dynamic behaviors shown of the chosen string, all figures correspond to screenshots of Matlab videos. These videos related to the string behavior can be seen in the String reproduction list.

Finally, the last part evaluated for the detailed string is the P-V-I curve that was also generated for the PV simplified simulation.

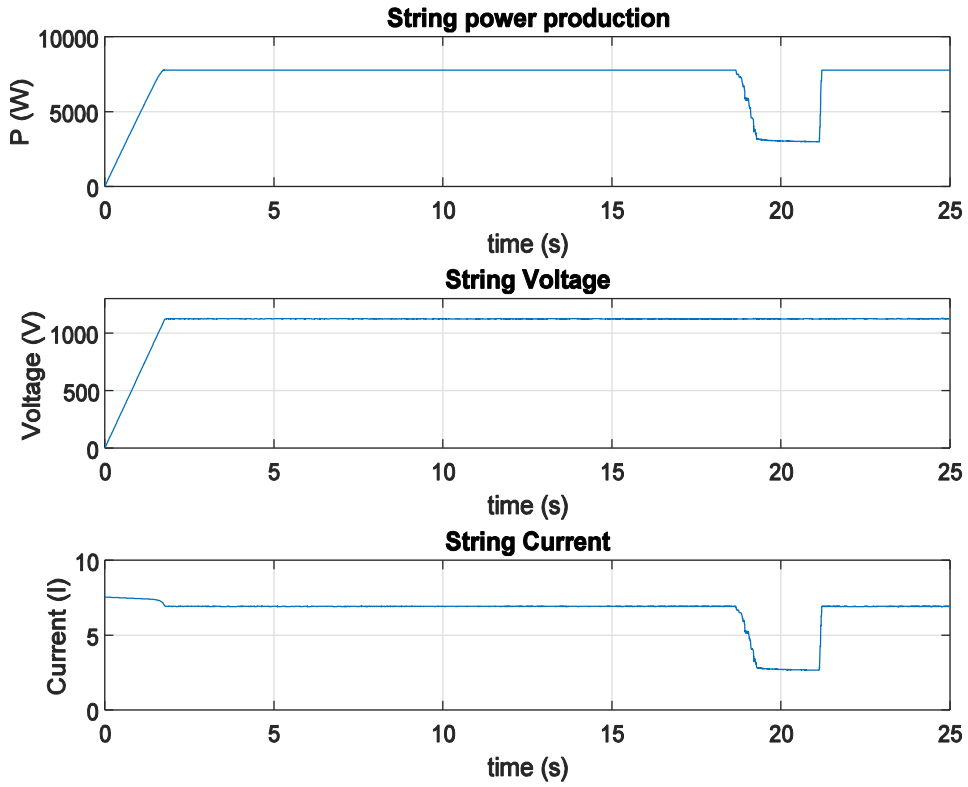


*Figure 42 String P-V-I MPPT measurements*

As shown in the other case, the first 2 seconds correspond to MPPT algorithm search. The difference between them is found at the current drop. In this case the current drop is adjusted to the string instead of the power plant, so the drop does not look so abrupt.

The following sequences have the same behavior than the behavior that happened in the general case. Nevertheless, it is clearly observed that at time sequence 18 seconds the string suffers a power drop. The power drop is produced because of the cloud shadowing.

Compared with Figure 34, in Figure 42 the path of the cloud is not reflected in their whole entrance. The reason is because it is analyzed one string and the entrance and the exit of the cloud in the power plant affects in a general way whereas in here the effect is evaluated to the string and the cloud effect become more notorious when it appears.

### 7.4.4 VSC simulation

The last part of the simulation as explained in other sections is referred to the VSC connected the grid source.

This simulation is fed with the overall current and voltage going out from the power plant, the grid constraints and the equipment between these two points.

When the grid network is connected to VSC is when it starts the process of the voltage source converter model.

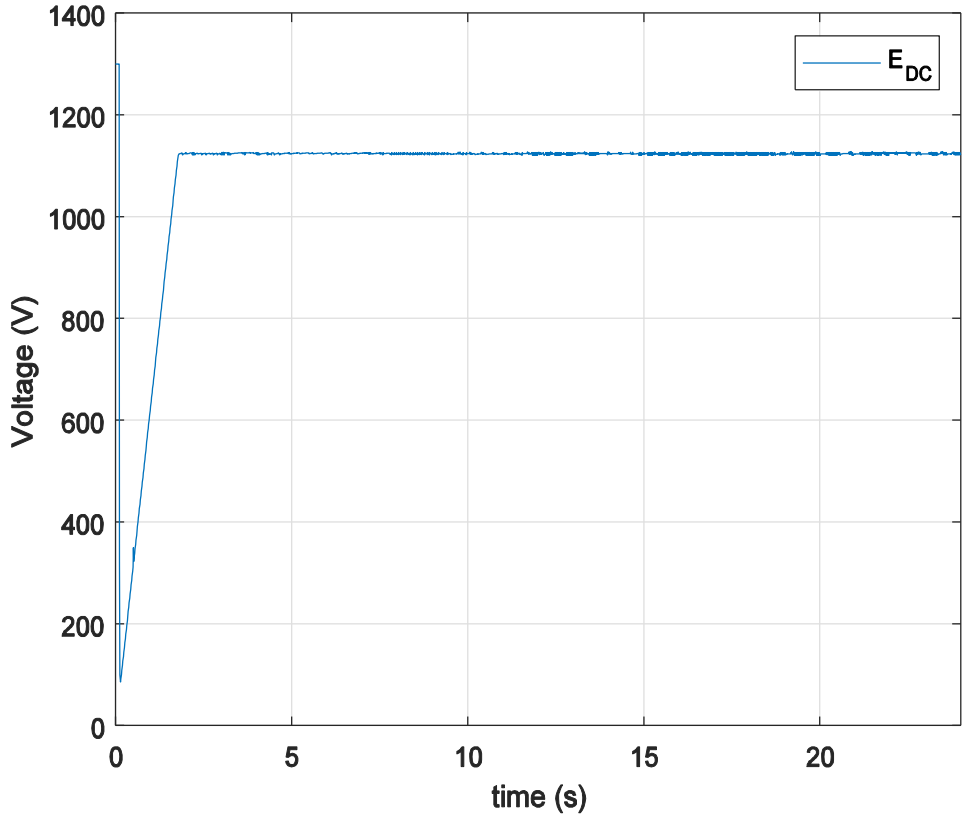


*Figure 43 $E_{DC}$*

As it can be seen in Figure 43, at the very beginning of the process the DC voltage of the DC bus suffers a transient because the starting voltage was initialized at 1300 V. After that it is readjusted to the voltage of the MPPT.

The produced ripple on the signal is the same as it is produced in the MPPT due to the algorithm.

The grid frequency behaves in a similar way it behaves the $E_{DC}$. It generates a really small transient at the beginning of the simulation but it is unappreciable during the whole simulation.

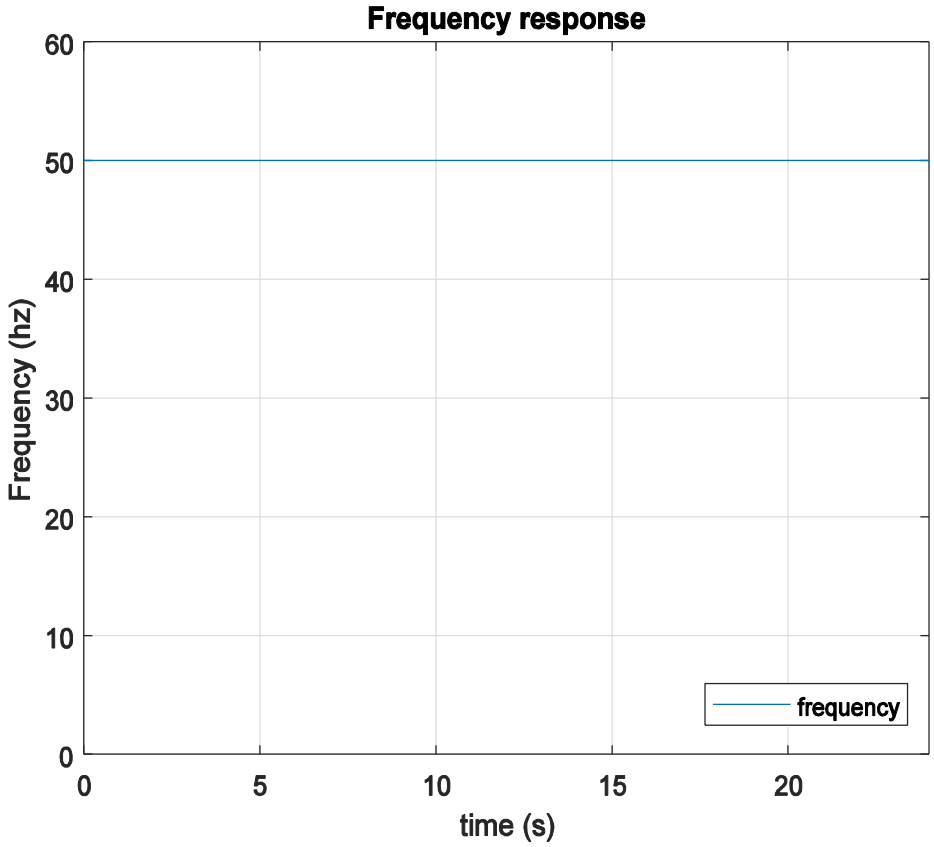


*Figure 44*

The grid frequency is considered at 50Hz and it can be observed from Figure 44 that this frequency is synchronized by the process carried by the phase locked loop.

As seen in Figure 45, the voltage of the VSC is a rippled voltage whereas the grid voltage is a line signal without the any ripple. It cannot be seen in the graphic but the voltage of the grid in the qd0 frame is behind the red one because the ripple hides the blue color.

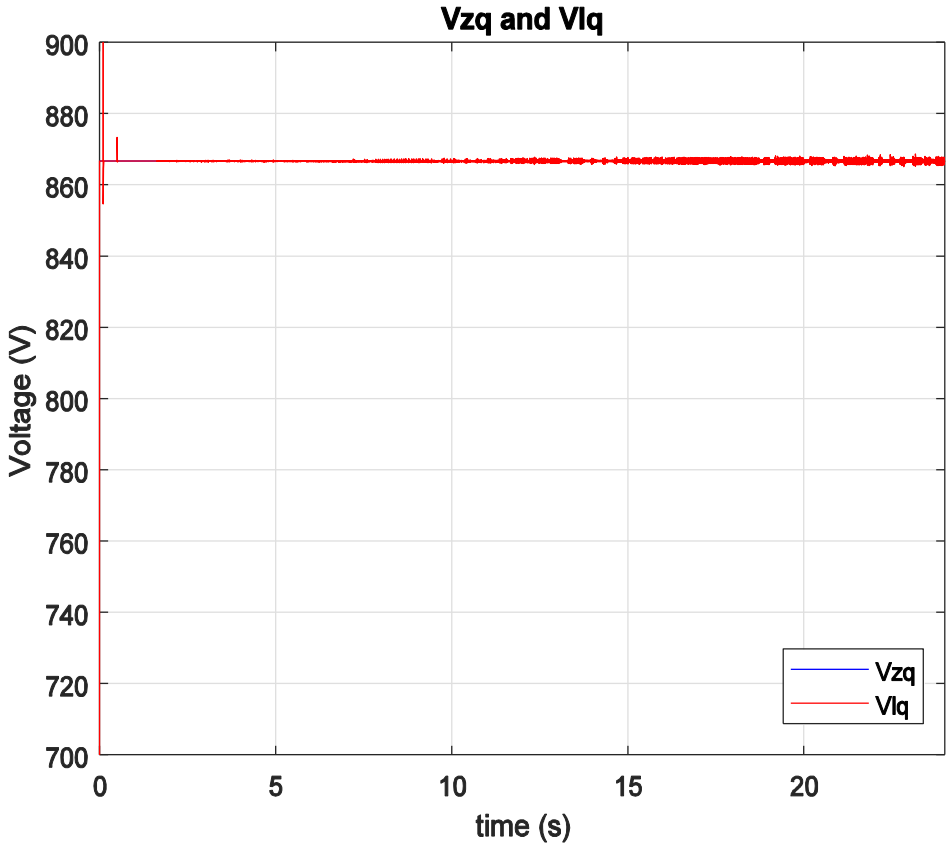


*Figure 45 Vzq and Vlq*

The reason why the red one is in a constant ripple is because the voltage coming from the MPPT is also a ripple. Therefore, the VSC is always readapting the voltage to the controlled one.

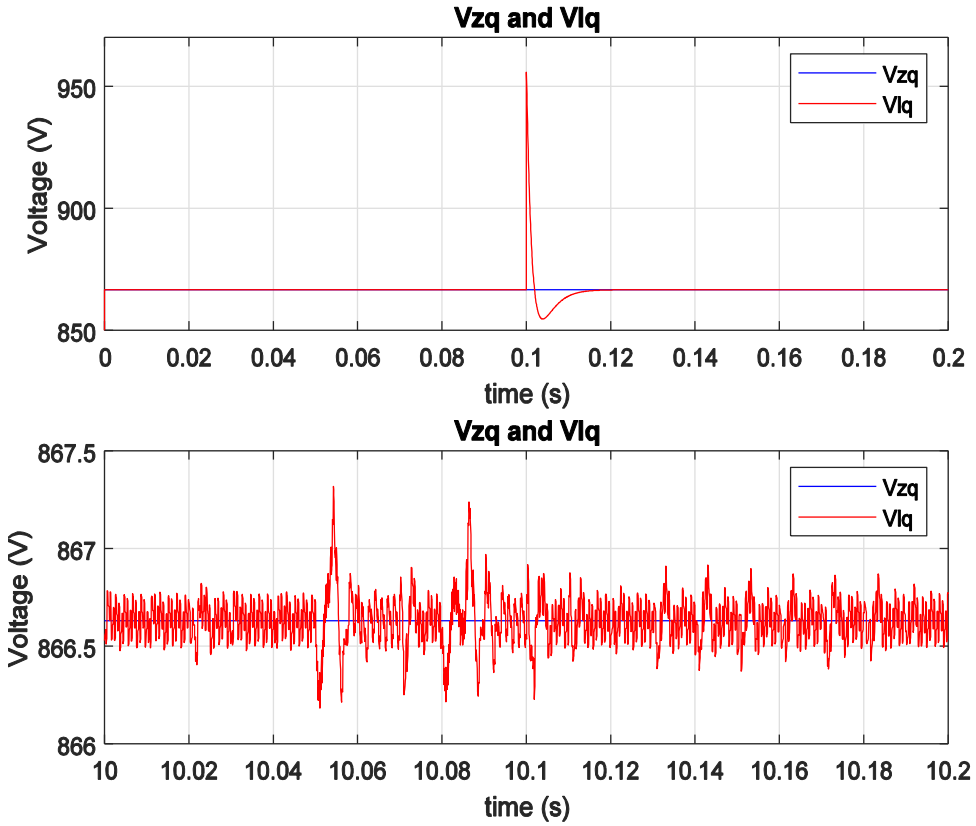


*Figure 46 Zoom in q-domain voltages*

When zooming in at some points like in Figure 46, it is observed how the voltages try to adjust to the grid voltage value. When focusing on the general case it is seen how the voltage produces ripples due to the MPPT and the voltage of the VSC still adapts to provide the grid voltage fed.

The same fact has to happen for the d-domain voltage in the qd0 frame. As there is not considered reactive power from the grid side the value d-domain voltage remains always at 0 V.

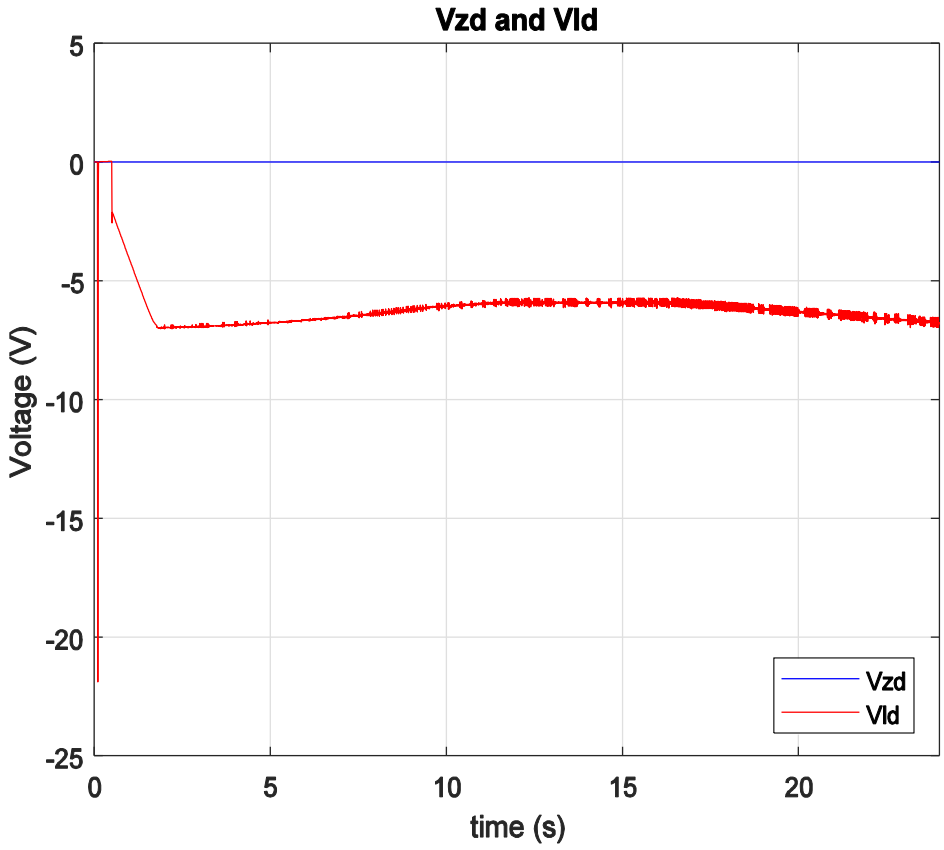


*Figure 47 Vzd and Vld*

Whereas the voltage in the VSC is always adapting because of the changes of the MPPT, the same ripple is produced because of the MPPT, as shown in Figure 48.

Additionally, it is observed that the d-domain voltage on the VSC side does not become negative until the VSC starts to inject power to the network.

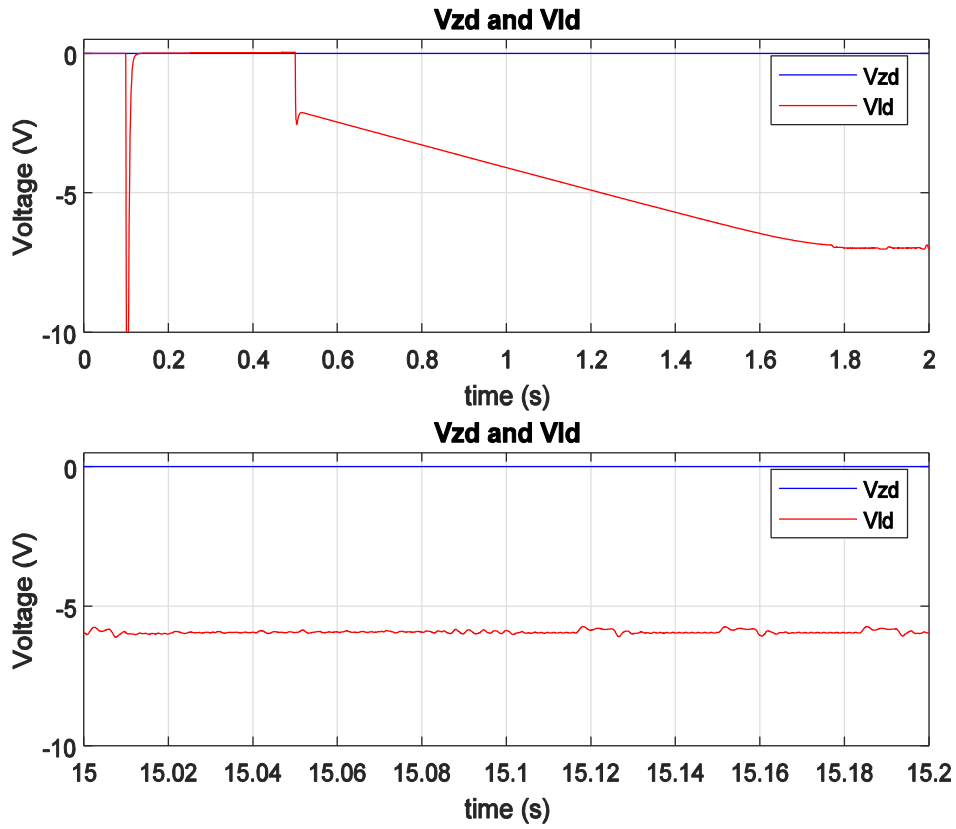


*Figure 48 Zoom in d-domain voltages*

Before finalizing with the voltages, these voltages in the qd0 domain are coupled and transformed into the abc domain. In Figure 49, a few times sequences are observed but differences are unappreciable. Just a sudden power change would produce s significant variation in the VSC side.

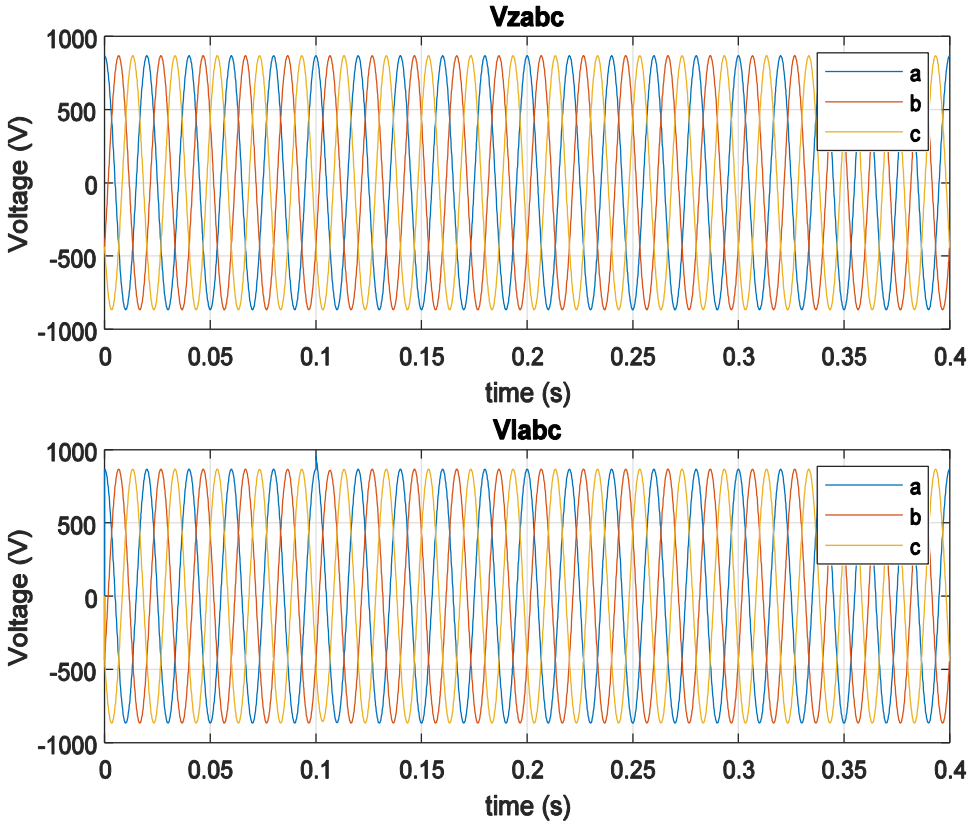


*Figure 49 Vzabc and Vlabc*

When addressing the focus on the qd0 domain currents it is observed that the currents should be the same and the output current of the VSC should be the same as the current of the grid side.

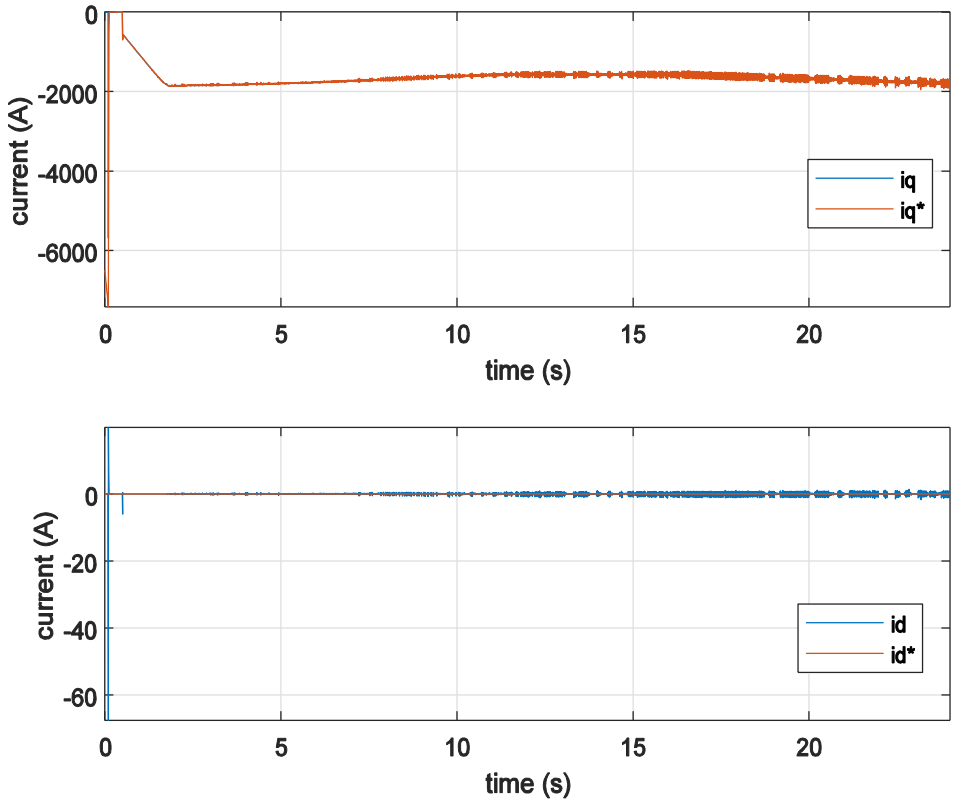


*Figure 50 Izqd and Ilqd*

As it is been happening in the voltage part in the current part also appears some ripples due to the MPPT. However, in a general way these ripples are quite low and the current going out of the VSC is adapting the grid current.

Figure 50 shows how close the ripple is to the grid value. Moreover, this graph shows the clearly that the VSC does not inject power to the grid until medium second later the simulation starts. This is produced expressly to allow the current control loop to compensate transients and make a fast adaptation to the grid. After, being started the power injection the currents follow the path adapting to the grid current.

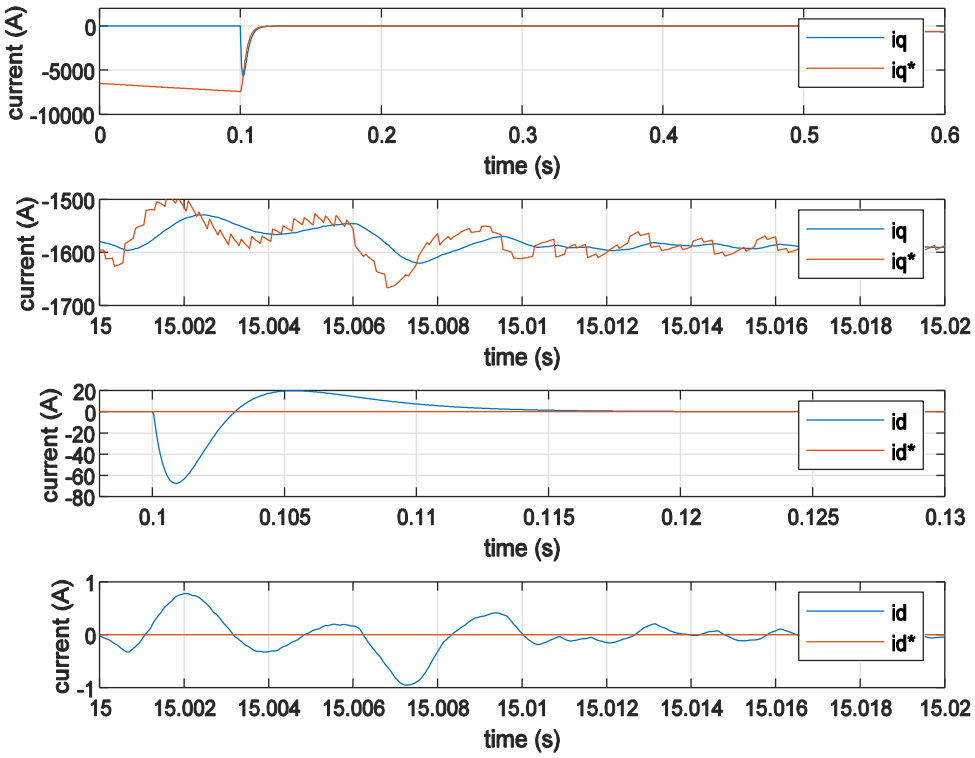


*Figure 51 Zoom in dq0 currents*

When analyzing in a deeper precision the ripples it is seen how the generated VSC current tries to copy the grid current by rippling the signal. Figure 51 shows in a zoomed precision what is happening in small time series.

Those times series that goes from the 0 seconds to almost 0,5 seconds have a better coupling than the others produced but these corresponds to zero injection that is the previous time to connect.

The other time series that goes from the 2 seconds to the end of the simulation is constantly producing the same signal in some kind of a delay and a ripple.

When these qd0 frame currents are transformed back to abc frame it is observed that these ripples seem to be disappeared.

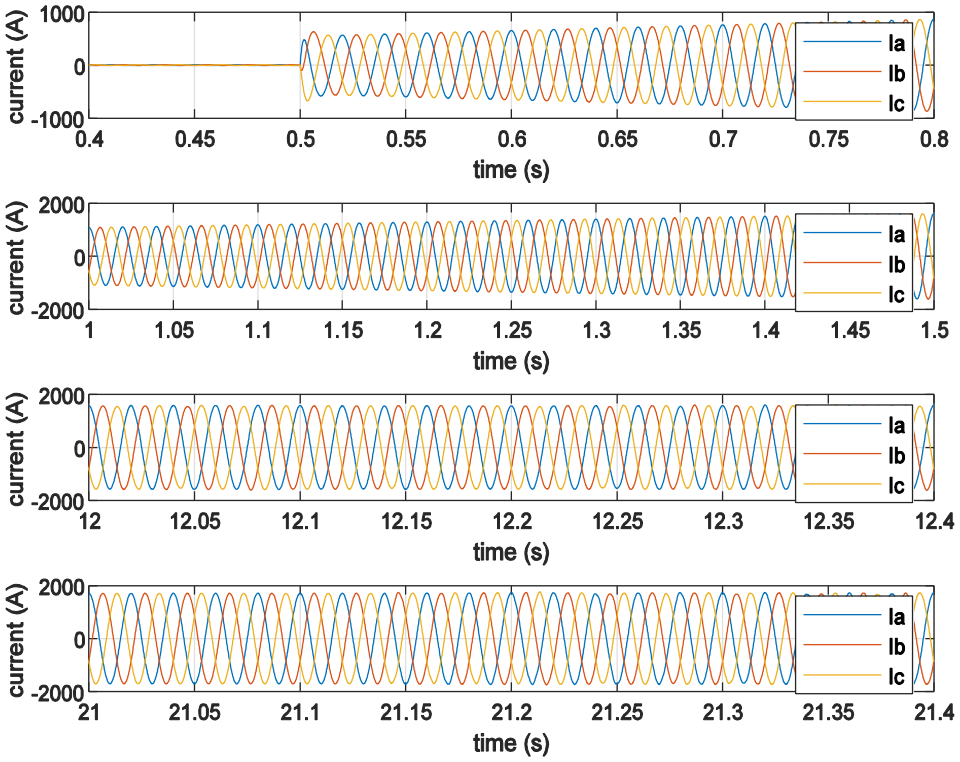


*Figure 52 Currents in the abc frame*

Different time series of the abc frame currents are projected in Figure 52, where in the first row it is observed the sudden change when starting to inject power.

In the second row, it is seen how it is increasing the generated power due to the MPPT tracking algorithm that increases from a low voltage in the DC side to higher voltages.

The last two rows are related to the current injection when the cloud is going through the solar field. However, no relevant changes can be observed. On the last row it seems that the injected current is slightly higher than the injected current on the third row.

Finally, the generated AC power injected to the grid is compared with the generated DC power produced in the DC side of the power plant.

In general terms it can be observed in Figure 53 that the power generated matches with the injected power. Despite the appearing ripple on the injected power, the inverter fulfills the functionality of transforming the DC power into AC power in the proposed control scheme. The methodology shows how the inverter controls its internal controllers in order to extract the maximum power of the DC side always taking into account the grid constraints.

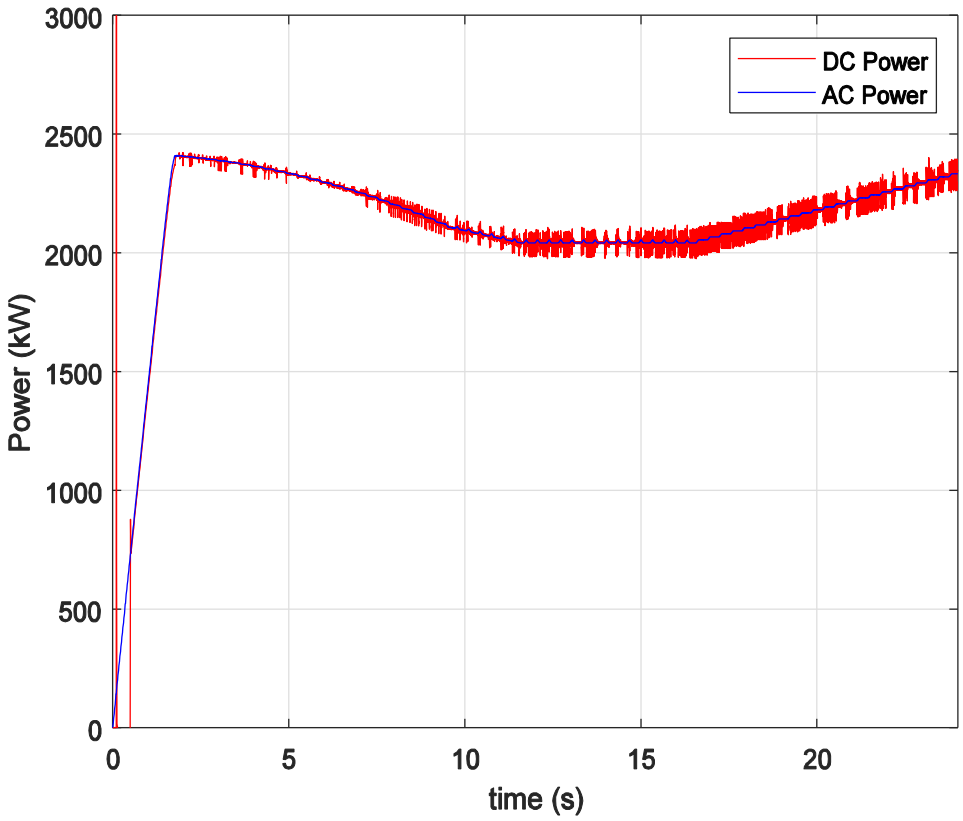


*Figure 53 AC and DC power*

# Conclusions

The general conclusion of the Project is that a trustable tool has been carried out successfully due to the exposed results corresponds with the expected results.

As the aim of the project was to simulate general results and provide a trustable tool to generate PV voltages and currents with a more trustable tool than the PV array of Simulink it can be said that the main target have been achieved according the created expectations.

However, it also has to be said that this tool has several limitations because it simulates scenarios at a really accurate detail. That is why the main simulation which is the simplified PV power generation takes between 2 and 3 hours to carry out a simulation of 25 seconds, being it the main disadvantage of the tool.

At the beginning of the programming and because of the algorithm loop generated in the PV module the time required to make the simulation was about 15 days. This is an unreasonable amount of time for simulation processes because ordinary computers like the one used to make this simulation cannot withstand this kind of requirements. After debugging the program with the commented low-pass filter solution qualitative gains were reached. Even so, the tool would need a little more debugging in order to reach a maximum of 10 minutes simulation for the 25 seconds of simulation.

This is also the main reason why 30 modules have been simplified into one module that represents the whole string. At the configuring screen of Simulink the generation of the full detailed model was crashing when a peak power of 100 kW was generated and the generate Simulink file was about 10Mb meaning that the entire file expected would have been about 300Mb. Hence, it was impossible to generate 3 MW.

Another important part not exposed at the targets is the data management. At the beginning it was not expected to work with a huge amount of data but one can see how much information this tool can provide. The key of being organized is really important because it could be messy to treat the amount of information that has been treated.

The last important trouble about the model is that it is not an automatic model. One cannot introduce the number of strings it is interested to study and, automatically the Simulink model is generated. It takes some time to do it manually which is an important drawback.

There are some following steps to make this tool more useful and profitable for different kind of companies and investigators. The first one is to debug those processes that take too much time to be carried out. The second step is to generate the whole tool like a mask

program in Simulink. The third step is to go deeper in the cloud analysis which means to take into account other surrounding environment conditions such as Albedo, cloud density, cloud dynamic shapes. The last implementation is on the grid connection, the model should be further developed allowing to study really complex cases involving black outs or some kind of grid perturbations in order to have the full tool ready.

After that, this tool could be suggested to other fields of PV to implement other kind of studies such as the predictable power generation of a power plant and hot spot detail analysis.

Additionally to the previous following steps, this model is a really visual tool meaning that it helps a lot to comprehend the real functioning of a PV power plant. It gives the chance to show how a PV power plant works in a graphical based case. Nevertheless, it has to be understood that some concepts require from more knowledge and comprehension.

# Project budget

This project cost depends mainly in the costs related to the analysis time spent on it, the material resources, the software, the energy consumed from the grid and Ethernet connection.

It is dedicated an amount of time equivalent to 1200 hours of work. This time takes in to account all activities that are performed through the thesis such as the simulation programming, the memory description, the dedicated time looking for information in sources like Reserchgate[19], IEEE Explore[20] and Science direct[21]. Moreover, this time counts on the time dedicated to learn new concepts of Matlab-Simulink and other easier software programs used in the whole project.

The simulation program counts on an annual license with some extensions. The main program is Matlab, which does not have the simulation blocks tool. The first extension is Simulink that consist on a simulating block tool. The other extension is Simulink electrical, which includes the electric blocks fully dedicated to the electric part.

The physical material necessary for this study consists on computer. As this is a theoretical project that is supported by simulations, it is not required more equipment than the mentioned.

This equipment needs to be fed with energy and Ethernet to work. These resources are also mandatory otherwise the project would get more difficulties to be made.

| Item | Quantity | Unit | Cost | Unit cost | Total cost |
|---|---|---|---|---|---|
| **Time** | 1200 | h | **20** | **€/h** | **24.000,00 €** |
| **Computer** | 1 | u | **1500** | **€** | **1.500,00 €** |
| **Matlab license** | 1 | u | **1200** | **€** | **1.200,00 €** |
| **Computer consumption** | 250 | W | **0,15** | **€/kWh** | **45,00 €** |
| **Ethernet** | 6 | month | **40** | **€/month** | **240,00 €** |
| **Lightning** | 10 | W | **0,15** | **€/kWh** | **1,80 €** |
| **Labour fees** | 1 | u | **4000** | **€** | **4.000,00 €** |
| **Total (without IVA)** | | | | | **30.986,80 €** |

*Table 7 Economic budget associated to the project*

As shown in Table 7, the cost of the project without governmental fees is calculated as

30.986,80 € (thirty thousand nine hundred eighty-six Euros and eighty cents).

| Item | Cost | Unit | IVA | Total cost |
|---|---|---|---|---|
| **Project** | 30986,8 | € | **21%** | **37.494,03 €** |

*Table 8 Overall project cost*

Finally, the cost of the project is calculated as 37.494,03 € (thirty-seven thousand four hundred ninety-four Euros and three cents).

# Environmental Impact

This project does not have any impact on the environment considering that it is a theoretical project and will not be implemented in a real project. However, one can consider the environmental impact of the resources used for the development of this thesis. Considering that this thesis took six months to be fully complete, working every day until reaching a total amount of 1200 hours. The electrical energy consumed by the computer and lighting of the office should be considered and it is shown in Table 9.

| Item | Power consumption (W) | Usage time (h) | Energy consumed (kWh) |
|---|---|---|---|
| **Computer** | 250 | 1200 | **300** |
| **Ethernet** | 10 | 1200 | **12** |
| **Lightning** | 10 | 1200 | **12** |
| **Total energy consumed (kWh)** | | | **324** |

*Table 9 Energy consumed for the project development*

According to REE [22], the mean grid green house gases emissions are 0,2628 kg$CO_2$/kWh.

| $CO_2$ Generated | Energy consumed (kWh) | Grid pollution (kg $CO_2$ / kWh) | CO2 emitted (kg CO2) |
|---|---|---|---|
| **Total $CO_2$** | 324 | 0,2628 | **85,1** |

*Table 10 Total amount of $CO_2$ produced*

# Bibliography

[1] T. Ackermann, G. Ran Andersson, and L. Sö Der A, “Distributed generation: a definition,” 2001.

[2] L. Premalatha and N. A. Rahim, “The Effect of Dynamic Weather Conditions on Three Types of PV Cell Technologies - A Comparative Analysis,” in *Energy Procedia*, 2017, vol. 117, pp. 275–282, doi: 10.1016/j.egypro.2017.05.132.

[3] “Simulink Documentation - MathWorks España.” [Online]. Available: https://es.mathworks.com/help/simulink/. [Accessed: 04-Apr-2020].

[4] “China’s JinkoSolar preserves its leading global solar PV module shipment rank in 2019,” *Global Data*, 2020. [Online]. Available: https://www.globaldata.com/chinas-jinkosolar-preserves-its-leading-global-solar-pv-module-shipment-rank-in-2019/. [Accessed: 21-Mar-2020].

[5] Krismadinata, N. A. Rahim, H. W. Ping, and J. Selvaraj, “Photovoltaic Module Modeling using Simulink/Matlab,” *Procedia Environ. Sci.*, vol. 17, pp. 537–546, Jan. 2013, doi: 10.1016/j.proenv.2013.02.069.

[6] X. H. Nguyen, “Matlab/Simulink Based Modeling to Study Effect of Partial Shadow on Solar Photovoltaic Array,” *Environ. Syst. Res.*, vol. 4, no. 1, pp. 1–10, Dec. 2015, doi: 10.1186/s40068-015-0042-1.

[7] A. Egea-Alvarez, A. Junyent-Ferré, and O. Gomis-Bellmunt, “Active and Reactive Power Control of Grid Connected Distributed Generation Systems,” *Green Energy Technol.*, vol. 96, pp. 47–81, 2012, doi: 10.1007/978-3-642-22904-6_3.

[8] A. E. García, A. Tutor, J. Ángel, and O. Ramírez, “Maximum Power Point Tracking Algorithms for Solar Photovoltaic Systems,” 2017.

[9] E. M. Subirón, “Desarrollo de una aplicación en Matlab para la evaluación de algoritmos MPPT y GMPPT,” Barcelona, 2016.

[10] R. Raedani and M. Hanif, “Design, testing and comparison of P&O, IC and VSSIR MPPT techniques,” in *3rd International Conference on Renewable Energy Research and Applications, ICRERA 2014*, 2014, pp. 322–330, doi: 10.1109/ICRERA.2014.7016404.

[11] A. EL, E. MEHDI, and M. ZAZI, “PSIM and MATLAB Co-Simulation of Photovoltaic System using ‘P and O’ and ‘Incremental Conductance’ MPPT,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 7, no. 8, 2016, doi: 10.14569/ijacsa.2016.070811.

[12] L. Norum, B. Hoff, and N. Mustafa, “Design of a Grid Connected Photovoltaic Power Electronic Converter,” UiT Norges arktiske universitet, Jun. 2017.

[13] L. Harnefors and H. P. Nee, “Model-based current control of ac machines using the internal model control method,” *IEEE Trans. Ind. Appl.*, vol. 34, no. 1, pp. 133–141, 1998, doi: 10.1109/28.658735.

[14] A. N. Varnavsky and V. M. Zhuravlev, “Simulation of timely warning of a driver about collision with oncoming vehicles using a car DVR,” in *2016 Dynamics of Systems, Mechanisms and Machines, Dynamics 2016*, 2017, doi: 10.1109/Dynamics.2016.7819104.

[15] C. Solar Inc, “Canadian Solar MaxPower CS6U-330P (1500V).”

[16] General Electric, “LV5 + Solar Inverter Data Sheet.”

[17] E. Molenbroek, D. W. Waddington, and K. A. Emery, “Hot spot susceptibility and testing of PV modules,” in *Conference Record of the IEEE Photovoltaic Specialists Conference*, 1992, vol. 1, pp. 547–552, doi: 10.1109/pvsc.1991.169273.

[18] “Mínimo, medio y máximo precio de la casación del mercado diario | OMIE.” [Online]. Available: https://www.omie.es/es/market-results/interannual/daily-market/daily-prices?scope=interannual. [Accessed: 18-Apr-2020].

[19] “ResearchGate | Find and share research.” [Online]. Available: https://www.researchgate.net/. [Accessed: 05-Apr-2020].

[20] “IEEE Xplore Digital Library.” [Online]. Available: https://ieeexplore.ieee.org/Xplore/home.jsp. [Accessed: 05-Apr-2020].

[21] “ScienceDirect.com | Science, health and medical journals, full text articles and books.” [Online]. Available: https://www.sciencedirect.com/. [Accessed: 05-Apr-2020].

[22] REE, “INVENTARIO DE EMISIONES DE CO 2 DE RED ELÉCTRICA DE ESPAÑA,.”

[23] G. Kruger, “PV array with algebraic loop broken,” *Mathworks*, 2019. [Online]. Available: https://es.mathworks.com/matlabcentral/fileexchange/71682-pv-array-with-algebraic-loop-broken. [Accessed: 05-Apr-2020].

[24] G. Kruger, “Real time tunable filters - File Exchange - MATLAB Central,” *Mathworks*, 2017. [Online]. Available: https://es.mathworks.com/matlabcentral/fileexchange/58498-real-time-tunable-filters. [Accessed: 05-Apr-2020].

# Annexes

# Annex A Matlab and Simulink Troubleshooting

Matlab and Simulink can present several problems when trying to simulate. This section is dedicated to those fundamental troubles that show how to solve the most important advertisings, warnings or errors that the software finds.

However, Matlab ans simulink count on a huge community that allows solving many of this kind of troubleshooting. Therefore, all complicated situations that are always solved somehow. There exist a huge variety of content that can be used to solve problems from other fields. Nonetheless, Matlab provides a functional guide for beginners [3].

## Simulink algorithm loops

The most peculiar issue found in the simulations is the algorithms loops. This warning is a simple warn about the existence of an algorithm loop in the program.

This algorithm loop occurs when a first output variable depends also on the output variable. The concept is to solve an equation like (54). It cannot be solved if an iterative process is applied.

$$x = sin(x) \tag{54}$$

It does not represent a problem at all for Simulink because they implemented their own solver when the program detects that there exist this kind of loop.

The real trouble with this Simulink solver is the time that takes to find the solution. In this particular case where a power plant of 3 MW is being simulated it represents a problem. The biggest issue is that the programmed scenario counts on a big amount of equations and blocks variables making the simulating file quite heavy.

In Figure 54 is shown the problematic loop scheme found in the PV module. It is seen how the voltage measured between both parts of the capacitor has to be reintroduced in the Matlab function to evaluate the recombination current of the diode, particularly this voltage is used to calculate the (7).

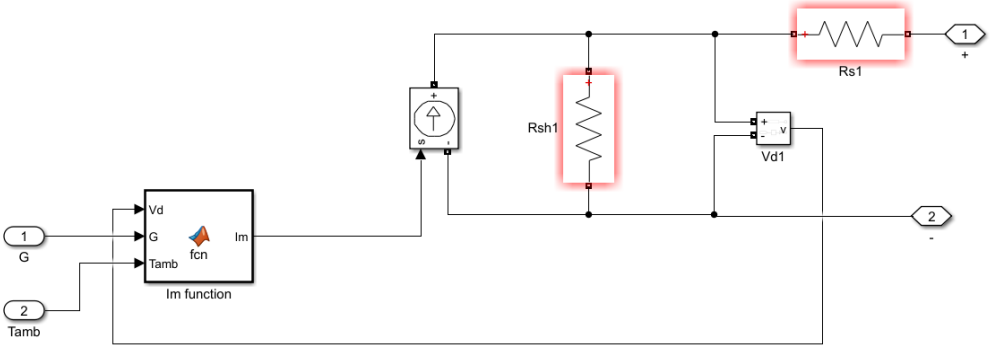


*Figure 54 Scheme of the containing algorithm loop*

When trying to solve this scenario Simulink takes up to 13 days to simulate a simulation of 25 seconds. Thus, a solution has to be suggested to debug this Simulink solver. As commented before this problem is produced as many times as modules, cells or detailed strings are in the simulation. All in all, this algorithm loop appears too much times so this solver process becomes tough and slow.

Then, a solution is provided so as not to invest an irrational amount of time trying to simulate with the Matlab solver. The solution applied is the usage of low-pass filters applied on the voltage, the power and the current measurement [23].

The usage of a low-pass filter allows not using the block memory that Simulink also counts on. The reason of not using the memory block, which could seem a good idea, is because it does not interpret well the functioning. When the voltage almost reaches it maximum power point the voltage measurement suddenly crushes and the whole simulation crushes at the same time, as well.

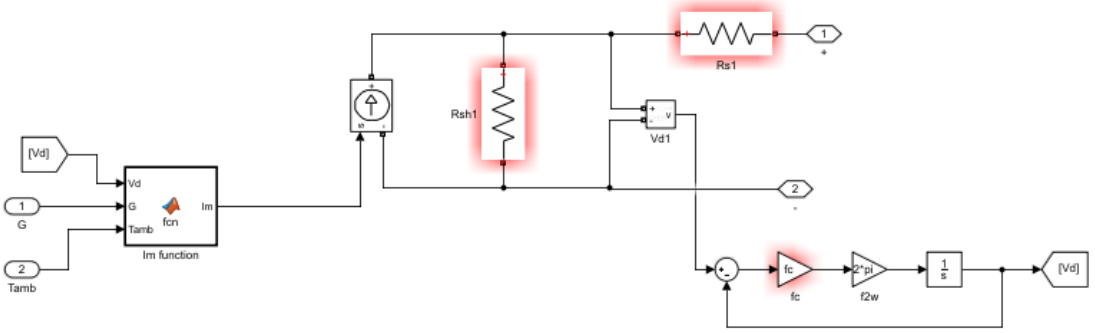


*Figure 55 Scheme of the module block applying low-pass filters*

Knowing that the simplest blocks from Simulink cannot be used, this low-pass provides the perfect decoupling to the solver without crashing or giving more problems.

The first low-pass filter is applied in the modules as shown in Figure 55. However, it also has to be used in other measurements otherwise the simulation crushes, as well.

In case of the other measurements such as current and voltage measurement of the whole power plant, the low-pass applied is form of a transfer function [24], as shown in (55).

$$F(s) = \frac{1}{s+1} \tag{55}$$

This transfer function breaks the algorithm loop allowing Simulink to simulate faster than it was doing with its own solver. The main function is like applying a memory block but shifting the time measurement.

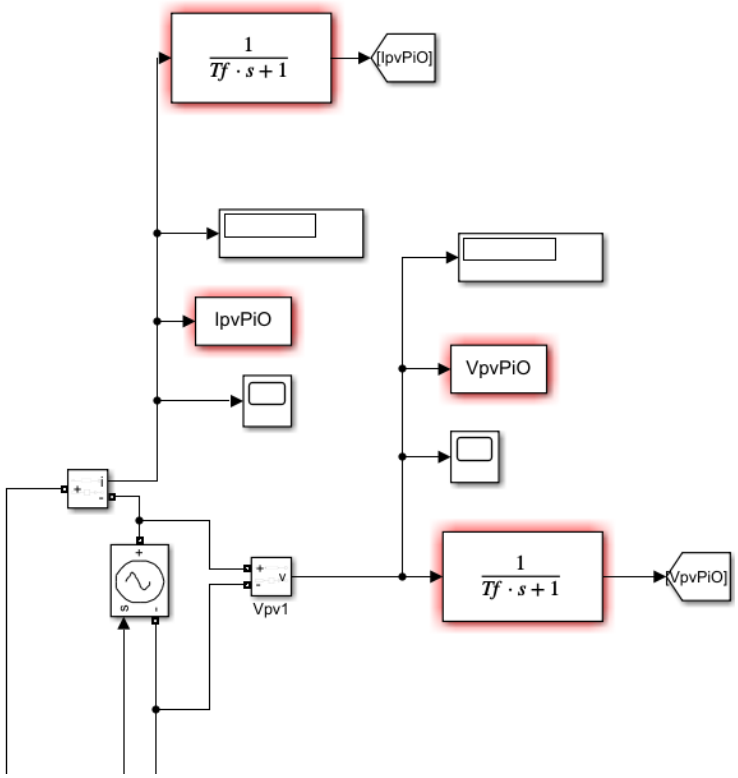


*Figure 56 Low-pass filters on the plant measurements*

This model is applied to every different case module that has been needed to analyze. The detailed string measurement and the general power plant simulation being the main simulation that require from this solution to work faster than what it was excepted with the own Simulink solver.

The reduction of the time has been decreased from 13 days to 2 hours and 30 minutes. It can be said that the solution is transcend. Nevertheless, the amount of time that the computer requires is still quite important for its simulation.

### Automatic input variable

A really jigsaw issue when trying to generate some heavy files in Simulink is to organize automatically and rename automatically the block names.

When one is repeating the same block function many times it also has to rename the block to the correspondent variable afterwards. Then, it is when someone realizes that changing manually all those variables become insane because it has to invest a n important amount of time by doing some stupid activity that could have been programmed to be done automatically.

This fact happens when copying blocks of Simulink such as “Go to” and “From”.

To rapidly solve this situation it is been generated a small code program in Matlab to do the rename automatically [5].

The proposed code is attached in Annex D Automatic input variable.

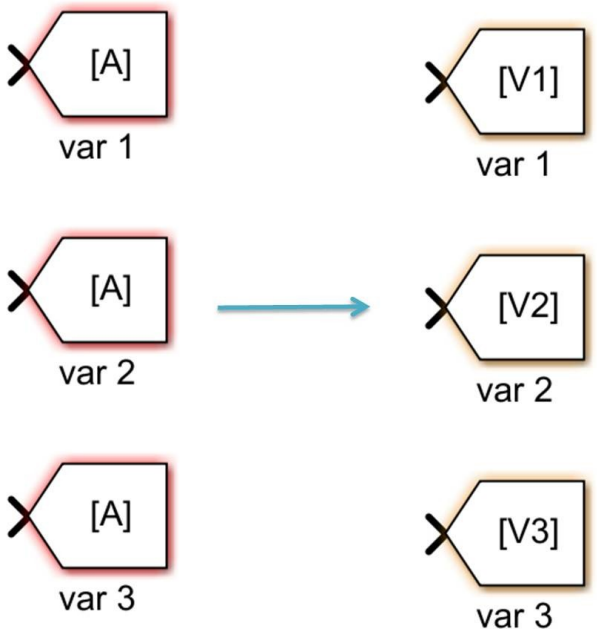


*Figure 57 Programmed solution*

The code will change automatically the values of the desired blocks because each block has a unique surname that when the block is copied changes its number name to the following number name.

Applying an iterative process for the casuistic explained solved the manual changing block name in seconds.

# Annex B Park transformation

The Park transformation is based on the Clark electrical transformation[7],[12]. In Clark, the transformation of three phase instantaneous electrical quantities expressed in the abc reference frame is transformed to a αβ0 orthogonal reference frame. Nonetheless, the Park transformation introduces the rotation variable in the Clark transformation.

$$[x_{\alpha\beta\,0}] = [T_{\alpha\beta\,0}][x_{abc}] \tag{56}$$

$$[T_{\alpha\beta\,0}] = \frac{2}{3}\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & -\frac{\sqrt{3}}{2} & \frac{\sqrt{3}}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \tag{57}$$

The inverse of the Clark transformation,

$$[x_{\mathrm{abc}}] = [T_{\alpha\beta\,0}]^{-1}[x_{\alpha\beta\,0}] \tag{58}$$

$$[T_{\alpha\beta\,0}]^{-1} = \frac{2}{3}\begin{bmatrix} 1 & 0 & 1 \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} & 1 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & 1 \end{bmatrix} \tag{59}$$

In the Park transformation, (57) and (59) are adjusted form the oscillating nature of the electric power system due to the interest in working with constant quantities.

$$[x_{\mathrm{qd}\,0}] = [T_{\alpha\beta\,0}][x_{abc}] \tag{60}$$

And its inverse

$$[x_{\mathrm{abc}}] = [T_{\alpha\beta\,0}]^{-1}[x_{\mathrm{qd}\,0}] \tag{61}$$

Finally, in the park transformation the transformation matrixes are written as

$$[T_{qd0}] = T(\theta) = \frac{2}{3}\begin{bmatrix} \cos(\theta) & \cos\left(\theta - \frac{2\pi}{3}\right) & \cos\left(\theta + \frac{2\pi}{3}\right) \\ \sin(\theta) & \sin\left(\theta - \frac{2\pi}{3}\right) & \sin\left(\theta + \frac{2\pi}{3}\right) \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \quad (62)$$

$$[T_{qd0}]^{-1} = T^{-1}(\theta) = \frac{2}{3}\begin{bmatrix} \cos(\theta) & \sin(\theta) & 1 \\ \cos\left(\theta - \frac{2\pi}{3}\right) & \sin\left(\theta - \frac{2\pi}{3}\right) & 1 \\ \cos\left(\theta + \frac{2\pi}{3}\right) & \sin\left(\theta + \frac{2\pi}{3}\right) & 1 \end{bmatrix} \quad (63)$$

The reasoning of the park transformation is as explained before to work with constant values that make the whole control easier then controlling the abc frame.

# Annex C Cloud dynamics

```
function Sfout = fcn(Sf,C,x,y,clx,cly,Sfx,Sfy)
xround=ceil(x);% round to the higher integer value of the X axis
yround=ceil(y);% round to the higher integer value of the Y axis
Sf0=Sf;%To restore the Solar field matrix
for a=1:1:clx
   Sfposx=xround-a+1;%Position of the solar field to be substituted in the X axis
   Cposx=clx-a+1;%Position of the Cloud to substitute in the solar field in the X axis
   for b=1:1:cly
    xprim=abs(x-xround);
    yprim=abs(y-yround);
    Sfposy=yround-b+1;%Position of the solar field to be substituted in the Y axis
    Cposy=cly-b+1;%Position of the Cloud to substitute in the solar field in the X axis
    if 1<=Sfposy && Sfposy<=Sfy && 1<=Sfposx && Sfposx<=Sfx
       Sf0(Sfposy,Sfposx)=C(Cposy,Cposx);
    end
    if 1<=Sfposy && Sfposy<=Sfy && 1<=Sfposx && Sfposx<=Sfx
      if a==1||a==clx||b==1||b==cly
       if a==1 && b==1
             Sf0(Sfposy,Sfposx)=C(Cposy,Cposx)*abs(1-xprim*yprim);
            elseif a==1 && b>1 && b~=cly
             Sf0(Sfposy,Sfposx)=C(Cposy,Cposx)*abs(1-xprim);
            elseif a==1 && b==cly
             Sf0(Sfposy,Sfposx)=C(Cposy,Cposx)*abs((1-xprim)*yprim);
            elseif a>1 && b==1 && a~=clx
             Sf0(Sfposy,Sfposx)=C(Cposy,Cposx)*abs(1-yprim);
            elseif a==clx && b==1
             Sf0(Sfposy,Sfposx)=C(Cposy,Cposx)*abs((1-yprim)*xprim);
            elseif a==clx && b>1 && b~=cly
             Sf0(Sfposy,Sfposx)=C(Cposy,Cposx)*xprim;
            elseif a~=clx && a>1 && b==cly
             Sf0(Sfposy,Sfposx)=C(Cposy,Cposx)*yprim;
         elseif a==clx && b==cly
            Sf0(Sfposy,Sfposx)=C(Cposy,Cposx)*yprim*xprim;
       end
      end
    end
   end
end
Sfout=Sf0;
end
```

# Annex D Automatic input variable

```
%Changing the variable of the simulink block
for ii=1:1:306
s1 = 'IrrString';
s2 = num2str(ii);
s = strcat(s1,s2)
foundBlock = find_system('Modulefield_v2', 'BlockType', 'Constant', 'Name',s)
objParams = get_param(foundBlock{1}, 'ObjectParameters')
aux=ii
val=strcat('M',s2)
set_param(foundBlock{1}, 'Value', val)
end

%Changing the variable of the simulink block "Goto"
for ii=1:1:306
s1 = 'IrrString';
s2 = num2str(ii);
s = strcat(s1,s2)
foundBlock = find_system('Importdata_trial', 'BlockType', 'Goto', 'Name',s)
objParams = get_param(foundBlock{1}, 'ObjectParameters')
aux=ii
val=strcat('M',s2)
set_param(foundBlock{1}, 'GoToTag', val)
end
```

# Annex E P&O MPPT Algorithm

```
function V= fcn(Vp,Ip)
% min and max value are used to limit duty between
V_min=0.1;
V_max=45.5*30;

persistent Pold Vold;
%persistent Vold Pold;
% persistent variable type can store the data
% we need the old data to obtain difference
% between old and new value
if isempty(Vold)
   Vold=0;
   Pold=0;
end
Pot=Vp*Ip;
Vstep=0.1;
if (Pot>Pold) && (Vp>Vold)
   V=Vp+Vstep;
elseif (Pot>Pold) && (Vp<Vold)
   V=Vp-Vstep;
elseif (Pot<Pold) && (Vp>Vold)
   V=Vp-Vstep;
elseif (Pot<Pold) && (Vp<Vold)
   V=Vp+Vstep;
else
   V=Vp;
end
if V >= V_max
   V=V_max;
elseif V<V_min
   V=V_min;
end
Pold=Pot;
Vold=Vp;
end
```

# Annex F Irradiance Search for detailed string simulation

```
function [M_timestep, ttt]= fcn(t_real,M_string,time)
M_aux=zeros(1,90);
persistent tt
if isempty(tt)
    tt=1;
end

while time(tt)<t_real
    tt=tt+1;
end
for nn=1:1:90
    M_aux(nn)=M_string(1,nn,tt);
end
M_timestep=M_aux;
ttt=tt;
end
```

# Annex G Voltage Search for detailed string simulation

```
function [V, ttt]= fcn(t_real,VpvPiO,t_Modulefield_v2)
persistent tt
if isempty(tt)
    tt=1;
end

if t_Modulefield_v2(tt,1)<t_real
    tt=tt+1;
end
V=VpvPiO(tt,1);
ttt=tt;
end
```

# Annex H VSC grid script

```
%% Invertergrid inputs
clc
clear all
close all
Vp=866.63;% Voltage peak value
fgrid=50; %Hz
rl=0;
l_l=(36e-6)/3;
% tau for PLL
Kp_pll=1;
Ki_pll=2;
% taus for current loop
tau=1e-3;%1ms
Kp=l_l/tau;
Ki=rl/tau;
%DC voltage regulator control
KpDC=5;
KiDC=10;
%Inverter features
C_DC=50e-3;
EDC=1300; %DC Voltage
Pn=2.74;%MWn
In=2646.2;%A nominal current
Pop=2.817603325;%MW operating power
Iop=2715.3;%A operating current
location='C:\Users\xavie\OneDrive\TFM\DATA2';
addpath(location);
load('IpvPiO');
load('VpvPiO');
%load('PpvPiO');
load('t_Modulefield_v2');
```

# Annex I General Script

```
%% Defining values of the PV panel from Canadian Solar
clc
clear all
close all
Rsh=200/3;
Rs=0.26/3;
Tc=0.001;
Tf=0.5*Tc;
fc=1/Tc;
location='C:\Users\xavie\Dropbox\MASTER\TFM\Simulations\Test modules';
addpath(location);
tic;
sim('PiOtest_v3')
tiempo= toc;
fprintf('El proceso ha tardado %f segundos', tiempo);
% https://blogs.mathworks.com/simulink/2015/07/18/why-you-should-never-break-an-algebraic-loop-with-with-a-memory-block/
% algorithm loop solution
%% Simplified module with filters
%Defining values of the PV panel from Canadian Solar in a simplified model
clc
clear all
close all
Rsh=200;
Rs=0.26;
Tc=0.00005;
Tf=2*Tc;
fc=1/Tc;
Vini=45.6;
% https://blogs.mathworks.com/simulink/2015/07/18/why-you-should-never-break-an-algebraic-loop-with-with-a-memory-block/
% algorithm loop solution
%% For simplification string modeling
clc
clear all
close all
Rsth=200*30;% the equivalent shunt resistance for a string
Rst=0.26*30;%the equivalent series resistance for a string
Tamb=-3.75;%°C
Tc=0.00005;
Tf=2*Tc;
fc=1/Tc;
location='C:\Users\xavie\Dropbox\MASTER\TFM\Simulations\Test modules\Data';
addpath(location);
load('M.mat');
load('t_IrradiationMatrix.mat');
load('irr_TF');
[a,b,c]=size(irr_Tf);
```

```
time=zeros(1,c);
for cc=1:1:c
    time(1,cc)=t(cc,1);
end
decimationModulefield=10;
Pdecimation=decimationModulefield;
Edecimation=decimationModulefield;
Vdecimation=decimationModulefield;
Idecimation=decimationModulefield;
IpvPiO_decimation=decimationModulefield;
VpvPiO_decimation=decimationModulefield;
PpvPiO_decimation=decimationModulefield;
t_Modulefield_v2_decimation=decimationModulefield;
%%  Run simulink process and acquaint the process time
tic;
sim('PiOtest_V2')
tiempo= toc;
fprintf('El proceso ha tardado %f segundos', tiempo);
%% Module field cloud and module field irradiance
%Defining the module field
irrmax=800; %W/m² maximim irradiance without any cloud
irrmin=300; %W/m² minimum irradiation with fully clouded
%Cloud
%clc
%clear all
%close all
phi=40;%angle of cloud entrance
x0=0;%position in the X axis of the cloud at time 0
y0=0;%position in the Y axis of the cloud at time 0
V=20;%inlet velocity input
Vx=V*cos(degtorad(phi));%velocity on the X axis
Vy=V*sin(degtorad(phi));%velocity on the Y axis
C=ones(150);%Cloud dimennsion
Sf=zeros(360,256);%Solar Field simensions
Sfrad=ones(360,256);%Solar Field simension
[cly,clx]=size(C);%measuring the size of the cloud to be analyzed
[Sfy,Sfx]=size(Sf);%measuring the size of the solar field to be analyzed
% Defining the solar field module coordinates
Tf=zeros(Sfy,Sfx);
for i=1:5:Sfx
    for j=1:1:Sfy
          Tf(j,i)=1;
    end
end
decimation=100;
Sfout_decimation=decimation;
Irrfield_decimation=decimation;
irr_Tf_decimation=decimation;
t_decimation_IrradiationMatrix=decimation;
%% Ploting the tracker layout
imagesc(Tf(:,:));
```

```
grid on
xlabel('Abscissa direction (m)')
ylabel('North direction (m)')
title('Tracker Layout');
colorbar
map=[1 1 1
     1 0 1];
colormap(map);
%% Ploting animated Cloud
vidObj=VideoWriter('Cloud.avi');%Change de name of the video to your purpose
vidObj.FrameRate=5; % frames per second, decrease the value to make videos larger
and increase to make videos shorter
open(vidObj)
figh=figure;
for k=1:10:length(t)
    %clf
    auxt=t(k);
    mesh(Tf,'edgecolor','r');
    grid on
    map=[1 1 1
        1 0 1];
    colormap(map);
    grid on
    xlabel('Abscissa direction (m)');
    ylabel('North direction (m)');
    title(['Cloud shadow at t = ',num2str(auxt),'seconds'])
    xlim([0 256]);
    ylim([0 360]);
    xticks(0:50:256)
    yticks(0:30:360)
    ax = gca;
    ax.YDir = 'normal';
    view(0,90);
    colorbar
    pause(0.1);
    hold on
    mesh(Sfout(:,:,k))
    hold off
    currFrame=getframe(figh,[0 0 560 415]);
    writeVideo(vidObj,currFrame);
end
close(vidObj);
%https://es.mathworks.com/videos/videowriter-object-in-r2010b-69020.html
%http://faculty.washington.edu/lum/EducationalVideoFiles/Matlab05/AnimationMatlab.m
%https://www.youtube.com/watch?v=3I1_5M7Okqo
%% Ploting animated Irradiance field
vidObj=VideoWriter('Irrfield.avi');%Change de name of the video to your purpose
vidObj.FrameRate=5; % frames per second, decrease the value to make videos larger
and increase to make videos shorter
open(vidObj)
figh=figure;
```

```
for k=1:10:length(t)
    %clf
    auxt=t(k);
    imagesc(Irrfield(:,:,k))
    grid on
    xlabel('Abscissa direction (m)');
    ylabel('North direction (m)');
    xlim([0 256]);
    ylim([0 360]);
    xticks(0:50:256)
    yticks(0:30:360)
    ax = gca;
    ax.YDir = 'normal';
    title(['Irradiation at ',num2str(auxt),' seconds'])
    colorbar
    caxis([0 1000]);
    c = colorbar;
    c.Label.String = 'W/m²';
    colormap(flipud(hot));
    pause(0.1);
    currFrame=getframe(figh,[0 0 560 415]);
    writeVideo(vidObj,currFrame);
end
close(vidObj);
%% Ploting animated Irradiance module field (it plots the irradiance on the modules)
vidObj=VideoWriter('irr_Tf.avi');%Change de name of the video to your purpose
vidObj.FrameRate=5; % frames per second, decrease the value to make videos larger
and increase to make videos shorter
open(vidObj)
figh=figure;
for k=1:10:length(t)
    %clf
    auxt=t(k);
    imagesc(irr_Tf(:,:,k))
    grid on
    xlabel('Abscissa direction (m)');
    ylabel('North direction (m)');
    xlim([0 256]);
    ylim([0 360]);
    xticks(0:50:256)
    yticks(0:30:360)
    ax = gca;
    ax.YDir = 'normal';
    title(['Irradiation on modules at ',num2str(auxt),' seconds'])
    colorbar
    caxis([0 1000]);
    c = colorbar;
    c.Label.String = 'W/m²';
    colormap(flipud(hot));
    pause(0.1);
    currFrame=getframe(figh,[0 0 560 415]);
```

```
    writeVideo(vidObj,currFrame);
end
close(vidObj);
%% irradiance entrances of the simplified module field
%Changing the variable of the simulink block
for ii=1:1:306
s1 = 'IrrString';
s2 = num2str(ii);
s = strcat(s1,s2)
foundBlock = find_system('Modulefield_v2', 'BlockType', 'Constant', 'Name',s)
objParams = get_param(foundBlock{1}, 'ObjectParameters')
aux=ii
val=strcat('M',s2)
set_param(foundBlock{1}, 'Value', val)
end
%% Irradiance entrances of the simplified module field
%Changing the variable of the simulink block "Goto"
for ii=1:1:306
s1 = 'IrrString';
s2 = num2str(ii);
s = strcat(s1,s2)
foundBlock = find_system('Importdata_trial', 'BlockType', 'Goto', 'Name',s)
objParams = get_param(foundBlock{1}, 'ObjectParameters')
aux=ii
val=strcat('M',s2)
set_param(foundBlock{1}, 'GoToTag', val)
end
%% Irradiance entrances of the simplified module field
%Changing the variable of the simulink block "F"
for ii=1:1:306
s1 = 'IRR';
s2 = num2str(ii);
s = strcat(s1,s2)
foundBlock = find_system('Modulefield_v2', 'BlockType', 'From', 'Name',s)
objParams = get_param(foundBlock{1}, 'ObjectParameters')
aux=ii
val=strcat('M',s2)
set_param(foundBlock{1}, 'GoToTag', val)
end

%% time generation
[a,b,c]=size(irr_Tf);
time=zeros(1,c);
for cc=1:1:c
    time(1,cc)=t(cc,1);
end
%% Irradiance input generation
[a,b,c]=size(irr_Tf);
%m=0;
%n=1;
%mm=0;
```

```
%string=1;
M=zeros(306,c);
for cc=1:1:c
    n=1;
    string=1;
    mm=0;
  while n<b
      suma=0;
      m=0;
        while m<60
        m=m+1;
        mm=mm+1;
        suma=suma+irr_Tf(mm,n,cc);
            if m==60
                M(string,cc)=suma/m;
                string=string+1;
            end
            if mm==a
                n=n+5;
                mm=0;
            end
        end
  end
end
%% Loading output data
clc
location='C:\Users\xavie\OneDrive\TFM\DATA2';
addpath(location);
load('E.mat');
load('I.mat');
load('P.mat');
load('V.mat');
load('t_Modulefield_v2.mat');
load('IpvPiO');
load('VpvPiO');
load('PpvPiO');
load('irr_TF');
%% Energy produced per each string
E=zeros(100144,306);
for tt=1:1:length(t_Modulefield_v2)
    for string=1:1:306
        if tt==1
            E(tt,string)=P(tt,string)*t_Modulefield_v2(tt,1);
        else
            E(tt,string)=P(tt,string)*(t_Modulefield_v2(tt,1)-t_Modulefield_v2(tt-1,1))+E(tt-
1,string);
        end
    end
end

%% Overall energy produced
```

```
Energy=0;
for string=1:1:306
    Energy=E(100144,string)/3600/1000+Energy;%in kWh
end

%% Generating output values from the simplified PV simulation to create matrixes and
videos
t_Modulefield=size(t_Modulefield_v2);
deci=10;%decimation to filter values in case the matrixes of time are really big
t_rel=floor(t_Modulefield(1)/deci);% rounding down so as not to crack the simulation
[a,b,c]=size(irr_Tf);
P_out=zeros(a,b,t_rel);
E_out=zeros(a,b,t_rel);
%V_out=zeros(a,b,t_rel);
I_out=zeros(a,b,t_rel);
t_out=zeros(1,t_rel);
for t_M=1:1:t_rel
    n=1;
    string=1;
    mm=0;
    t_aux=t_M*deci;
    t_out(1,t_M)=t_Modulefield_v2(t_aux);
  while n<b
      suma=0;
      m=0;
        while m<60
        m=m+1;
        mm=mm+1;
        t_mes=t_M*deci;
        P_out(mm,n,t_M)=P(t_mes,string)/1000; % in kW
        E_out(mm,n,t_M)=E(t_mes,string)/3600; %in Wh
        %V_out(mm,n,t_M)=V(t_mes,string);
        I_out(mm,n,t_M)=I(t_mes,string);
            if m==60
                string=string+1;
            end
            if mm==a
                n=n+5;
                mm=0;
            end
        end
  end
end
%% Module temperature in the in the trakcer layout
Tamb=-3.75;%°C
NOCT=43;%°C
Tplant=(irr_Tf/800)*(NOCT-20)+Tamb;
vidObj=VideoWriter('Temperature modules power plant.avi');%Change de name of the
video to your purpose
vidObj.FrameRate=5; % frames per second, decrease the value to make videos larger
and increase to make videos shorter
```

```
open(vidObj)
figh=figure;
for k=1:10:length(t)
    time=t(k);
    imagesc(Tplant(:,:,k));
    grid on
    xlabel('Abscissa direction (m)');
    ylabel('North direction (m)');
    xlim([0 256]);
    ylim([0 360]);
    xticks(0:50:256)
    yticks(0:30:360)
    ax = gca;
    ax.YDir = 'normal';
    title(['Module temperature at ',num2str(time),' seconds'])
    colorbar
    caxis([0 30]);
    c = colorbar;
    c.Label.String = '°C';
    colormap(flipud(hot));
    pause(0.1);
    currFrame=getframe(figh,[0 0 560 415]);
    writeVideo(vidObj,currFrame);
end
close(vidObj);

%% Ploting animated power output
vidObj=VideoWriter('P_out.avi');%Change de name of the video to your purpose
vidObj.FrameRate=5; % frames per second, decrease the value to make videos larger
and increase to make videos shorter
open(vidObj)
figh=figure;
for k=1:100:length(t_out)
    auxt=t_out(k);
    imagesc(P_out(:,:,k))
    grid on
    xlabel('Abscissa direction (m)');
    ylabel('North direction (m)');
    xlim([0 256]);
    ylim([0 360]);
    xticks(0:50:256)
    yticks(0:30:360)
    ax = gca;
    ax.YDir = 'normal';
    title(['Power at ',num2str(auxt),' seconds'])
    colorbar
    caxis([0 10]);
    c = colorbar;
    c.Label.String = 'kW';
    colormap(flipud(hot));
    pause(0.1);
```

```
    currFrame=getframe(figh,[0 0 560 415]);
    writeVideo(vidObj,currFrame);
end
close(vidObj);

%% Ploting animated energy output
vidObj=VideoWriter('E_out.avi');%Change de name of the video to your purpose
vidObj.FrameRate=5; % frames per second, decrease the value to make videos larger
and increase to make videos shorter
open(vidObj)
figh=figure;
for k=1:100:length(t_out)
    auxt=t_out(k);
    imagesc(E_out(:,:,k))
    grid on
    xlabel('Abscissa direction (m)');
    ylabel('North direction (m)');
    xlim([0 256]);
    ylim([0 360]);
    xticks(0:50:256)
    yticks(0:30:360)
    ax = gca;
    ax.YDir = 'normal';
    title(['Energy produced per string at ',num2str(auxt),' seconds'])
    colorbar
    caxis([0 60]);
    c = colorbar;
    c.Label.String = 'Wh';
    colormap(flipud(hot));
    pause(0.1);
    currFrame=getframe(figh,[0 0 560 415]);
    writeVideo(vidObj,currFrame);
end
close(vidObj);

%% Irradiance entrances for detailed string
%Changing the variable of the simulink block "F"
for ii=1:1:90
s1 = 'IRRO';
s2 = num2str(ii);
s = strcat(s1,s2)
foundBlock = find_system('DetailedString', 'BlockType', 'Goto', 'Name',s)
objParams = get_param(foundBlock{1}, 'ObjectParameters')
aux=ii
val=strcat('M',s2)
set_param(foundBlock{1}, 'GoToTag', val)
end
%% Power output from the detailed modules
%Changing the variable of the simulink block "Goto"
for ii=1:1:30
s1 = 'PO';
```

```
s2 = num2str(ii);
s = strcat(s1,s2)
foundBlock = find_system('DetailedString', 'BlockType', 'From', 'Name',s)
objParams = get_param(foundBlock{1}, 'ObjectParameters')
aux=ii
val=strcat('P',s2)
set_param(foundBlock{1}, 'GoToTag', val)
end
%% Energy output from the detailed modules
%Changing the variable of the simulink block "Goto"
for ii=1:1:30
s1 = 'EO';
s2 = num2str(ii);
s = strcat(s1,s2)
foundBlock = find_system('DetailedString', 'BlockType', 'From', 'Name',s)
objParams = get_param(foundBlock{1}, 'ObjectParameters')
aux=ii
val=strcat('E',s2)
set_param(foundBlock{1}, 'GoToTag', val)
end
%% Detailed string generartion
clc
close all
clear all
nstring=209;%chose the string to be studied
column=ceil(nstring/6);
if column==1
    colfield=1;
else
    colfield=column*5+1;
end
rowaux=round((((nstring/6)-floor(nstring/6))*6),0);
if rowaux==0
    row=6;
else
row=round((((nstring/6)-floor(nstring/6))*6),0);
end
Tamb=-3.75;%°C
Rsh=200/3;
Rs=0.26/3;
Tc=0.001;
Tf=Tc;
%Tc=0.00005;
%Tf=2*Tc;
fc=1/Tc;
location='C:\Users\xavie\Dropbox\MASTER\TFM\Simulations\Test modules\Data';
addpath(location);
load('t_IrradiationMatrix.mat');
load('irr_TF');
[a,b,c]=size(irr_Tf);
time=zeros(1,c);
```

```
for cc=1:1:c
    time(1,cc)=t(cc,1);
end
M_string=zeros(1,90,c);
for cc=1:1:c
    startrow=(row-1)*60+1;
    lastrow=row*60;
    steprow=1;
    step=0;
    for irrin=1:1:90
        currstep=step;
        nextstep=step+(2/3);
        evalstep=currstep+nextstep;
        if ceil(evalstep)==floor(evalstep)
            first=floor(currstep);
            second=floor(nextstep);
            rowfield1=startrow+first;
            rowfield2=startrow+second;
            irr_in=(irr_Tf(rowfield1,colfield,cc)+irr_Tf(rowfield2,colfield,cc))/2;
            M_string(1,irrin,cc)=irr_in;
        elseif ceil(evalstep)~=floor(evalstep)
            rowfield3=floor(currstep)+startrow;
            M_string(1,irrin,cc)=irr_Tf(rowfield3,colfield,cc);
        end
        step=step+(2/3);
    end
end
Irrstring=zeros(1,60,c);
for cc2=1:1:c
    startrow=(row-1)*60+1;
    lastrow=row*60;
    steprow=1;
    step=0;
    for irrin=1:1:60
        nextstep=step+1;
        rowfieldcc2=startrow+nextstep;
        Irrstring(1,irrin,cc2)=irr_Tf(rowfieldcc2,colfield,cc2);
        step=step+1;
    end
end
location2='C:\Users\xavie\OneDrive\TFM\DATA2';
addpath(location2);
load('VpvPiO');
load('t_Modulefield_v2');
decimationModulefield=1;
Impp_string_decimation=decimationModulefield;
Vmpp_string_decimation=decimationModulefield;
Pmpp_string_decimation=decimationModulefield;
t_DetailedString_decimation=decimationModulefield;
Decimation_P_Detailedstrung_module=decimationModulefield;
Decimation_E_Detailedstrung_module=decimationModulefield;
```

```
location3='C:\Users\xavie\Dropbox\MASTER\TFM\Simulations\Test modules';
addpath(location3);
tic;
sim('DetailedString')
tiempo= toc;
fprintf('El proceso ha tardado %f segundos', tiempo);
%% Detailed string irradiation video
vidObj=VideoWriter('Irr_string.avi');%Change de name of the video to your purpose
vidObj.FrameRate=5; % frames per second, decrease the value to make videos larger
and increase to make videos shorter
open(vidObj)
figh=figure;
for k=1:10:length(t)
    %clf
    auxt=t(k,1);
    imagesc(Irrstring(:,:,k))
    grid on
    xlabel('Module witdh (m)');
    ylabel('string length (m)');
    ax = gca;
    ax.YDir = 'normal';
    axis equal
    xlim([0.5 1.5]);
    ylim([0 60]);
    xticks(1:1:1)
    yticks(0:2:60)
    title(['Module Irradiation at ',num2str(auxt),' seconds'])
    colorbar
    caxis([0 1000]);
    c = colorbar;
    c.Label.String = 'W/m²';
    colormap(flipud(hot));
    pause(0.05);
    currFrame=getframe(figh,[0 0 560 415]);
    writeVideo(vidObj,currFrame);
end
close(vidObj);

%% Energy produced per each module in the string
E_string=zeros(252342,30);
for ttt=1:1:length(t_DetailedString)
    for mod=1:30
        if ttt==1
            E_string(ttt,mod)=P_Detailedstrung_module(ttt,mod)*t_DetailedString(ttt,1);
        else
            E_string(ttt,mod)=P_Detailedstrung_module(ttt,mod)*(t_DetailedString(ttt,1)-
t_DetailedString(ttt-1,1))+E_string(ttt-1,mod);
        end
    end
end
```

```
%% Overall energy produced in the chosen string
Estring=0;
for mod=1:1:30
    Estring=E_string(252342,mod)/3600+Estring;%in Wh
end

%% Detailed power production per module on the detailed string
[tst,tr]=size(t_DetailedString);
divisiontime=100;
tst2=floor(tst/divisiontime);
P_outstring=zeros(60,tst2);
E_outstring=zeros(60,tst2);
t_outstring=zeros(tst2,1);
tit=0;
for ttime=1:1:tst2
    base=0;
    mod=1;
    for out=1:1:60
        base=base+1;
        if base==2
            P_outstring(out,ttime)=P_Detailedstrung_module(ttime*divisiontime,mod);
            E_outstring(out,ttime)=E_string(ttime*divisiontime,mod)/3600;% in Wh
            mod=mod+1;
            base=0;
        else
            P_outstring(out,ttime)=P_Detailedstrung_module(ttime*divisiontime,mod);
            E_outstring(out,ttime)=E_string(ttime*divisiontime,mod)/3600;% in Wh
        end
    end
    t_outstring(ttime,1)=t_DetailedString(ttime*divisiontime,1);
end

%% Detailed module power production in a chosen string
vidObj=VideoWriter('P_string.avi');%Change de name of the video to your purpose
vidObj.FrameRate=5; % frames per second, decrease the value to make videos larger
and increase to make videos shorter
open(vidObj)
figh=figure;
for k=1:10:length(t_outstring)
    %clf
    auxt=t_outstring(k,1);
    imagesc(P_outstring(:,k))
    grid on
    ylabel('string length (m)');
    ax = gca;
    ax.YDir = 'normal';
    axis equal
    xlim([0.5 1.5]);
    ylim([0 60]);
    xticks(1:1:1)
    yticks(0:2:60)
```

```
    title(['Module Power production at ',num2str(auxt),' seconds'])
    xlabel('Module witdh (m)');
    colorbar
    caxis([0 330]);
    c = colorbar;
    c.Label.String = 'W';
    colormap(flipud(hot));
    pause(0.05);
    currFrame=getframe(figh,[0 0 560 415]);
    writeVideo(vidObj,currFrame);
end
close(vidObj);

%% Energy production per module in a chosen string
vidObj=VideoWriter('E_string.avi');%Change de name of the video to your purpose
vidObj.FrameRate=5; % frames per second, decrease the value to make videos larger
and increase to make videos shorter
open(vidObj)
figh=figure;
for k=1:10:length(t_outstring)
    %clf
    auxt=t_outstring(k);
    imagesc(E_outstring(:,k))
    grid on
    xlabel('Module witdh (m)');
    ylabel('string length (m)');
    ax = gca;
    ax.YDir = 'normal';
    axis equal
    xlim([0.5 1.5]);
    ylim([0 60]);
    xticks(1:1:1)
    yticks(0:2:60)
    title(['Cumulative module Energy produced at ',num2str(auxt),' seconds'])
    colorbar
    caxis([0 2]);
    c = colorbar;
    c.Label.String = 'Wh';
    colormap(flipud(hot));
    pause(0.05);
    currFrame=getframe(figh,[0 0 560 415]);
    writeVideo(vidObj,currFrame);
end
close(vidObj);

%% Module temperature per string
Tamb=-3.75;%°C
NOCT=43;%°C
Tstring=(Irrstring/800)*(NOCT-20)+Tamb;
vidObj=VideoWriter('T_string.avi');%Change de name of the video to your purpose
```

```
vidObj.FrameRate=5; % frames per second, decrease the value to make videos larger
and increase to make videos shorter
open(vidObj)
figh=figure;
for k=1:10:length(t)
    %clf
    auxt=t(k,1);
    imagesc(Tstring(:,k))
    grid on
    xlabel('Module witdh (m)');
    ylabel('string length (m)');
    ax = gca;
    ax.YDir = 'normal';
    axis equal
    xlim([0.5 1.5]);
    ylim([0 60]);
    xticks(1:1:1)
    yticks(0:2:60)
    title(['Module temperature at ',num2str(auxt),' seconds'])
    colorbar
    caxis([0 30]);
    c = colorbar;
    c.Label.String = '°C';
    colormap('parula');
    pause(0.05);
    currFrame=getframe(figh,[0 0 560 415]);
    writeVideo(vidObj,currFrame);
end
close(vidObj);


%% Irradiation on the specific string to 60 matrix values
clc
close all
clear all
nstring=209;%chosen the string to be studied
column=ceil(nstring/6);
if column==1
    colfield=1;
else
    colfield=column*5+1;
end
rowaux=round((((nstring/6)-floor(nstring/6))*6),0);
if rowaux==0
    row=6;
else
row=round((((nstring/6)-floor(nstring/6))*6),0);
end
location='C:\Users\xavie\Dropbox\MASTER\TFM\Simulations\Test modules\Data';
addpath(location);
load('t_IrradiationMatrix.mat');
```

```
load('irr_TF');
[a,b,c]=size(irr_Tf);
time=zeros(1,c);
for cc=1:1:c
    time(1,cc)=t(cc,1);
end
Irrstring=zeros(60,1,c);
for cc2=1:1:c
    startrow=(row-1)*60+1;
    lastrow=row*60;
    steprow=1;
    step=0;
    for irrin=1:1:60
        nextstep=step+1;
        rowfieldcc2=startrow+nextstep;
        Irrstring(irrin,1,cc2)=irr_Tf(rowfieldcc2,colfield,cc2);
        step=step+1;
    end
end
```

# Annex J Graphics generation

```
%% Loading data
clc
close all
clear all
addpath('C:\Users\xavie\OneDrive\TFM\DATA2');
% Tracker layout
load('Tf.mat');

% from the irradiation matrix simulation
load('t_IrradiationMatrix.mat');
load('X.mat');
load('Y.mat');
load('Sfout.mat');
load('Irrfield.mat');
load('irr_Tf.mat');
load('X.mat');

% from the simplified pv simulation
load('t_Modulefield_v2.mat');
load('IpvPiO.mat');
load('VpvPiO.mat');
load('PpvPiO.mat');
load('P.mat');
load('I.mat');
load('E_out.mat');
load('P_out.mat');

%from a given time 14 seconds of the simulation I-V Curve
load('V_IVCurve.mat');
load('I_IVCurve.mat');
load('P_IVCurve.mat');
load('V_IVCurve_modules.mat');
load('I_IVCurve_modules.mat');
load('P_IVCurve_modules.mat');
load('t_IVCurve.mat');

%from the detailed string simulation
load('t_DetailedString.mat');
load('Impp_string.mat');
%load('Vmpp_string.mat');% not saved
load('Pmpp_string.mat');% not saved
load('P_Detailedstrung_module.mat');
load('Irrstring.mat');
load('P_outstring.mat');

%%
load('E_outstring.mat');
```

```
%%
load('t_outstring.mat');

%from the inverter grid simulation
load('w_n.mat');
load('Vzq.mat');
load('Vzd.mat');
load('Vzabc.mat');
load('Vlq.mat');
load('Vlout.mat');
load('Vld.mat');
load('Vlabc.mat');
load('t_invertergrid.mat');
load('P_AC.mat');
load('Izabc.mat');
load('Iqref.mat');
load('Iq.mat');
load('Ilabc.mat');
load('Idref.mat');
load('Id.mat');
load('I_DC_IN.mat');
load('f.mat');
load('E_DC.mat');

%% ploted time simulation
ts=1401;
ts2=5606;

%% Cloud path
imagesc(Tf(:,:));
map=[1 1 1
    1 0 1];
colormap(map);
grid on
xlabel('Abscissa direction (m)');
ylabel('North direction (m)');
title('Cloud Path');
%axis equal
xlim([0 256]);
ylim([0 360]);
xticks(0:50:256)
yticks(0:30:360)
ax = gca;
ax.YDir = 'normal';
hold on
plot(x,y,'b');
hold off
%% Cloud shadow
mesh(Tf,'edgecolor','r');
map=[1 1 1
    1 0 1];
```

```
colormap(map);
grid on
xlabel('Abscissa direction (m)');
ylabel('North direction (m)');
time=t(ts);
%axis equal
title(['Cloud Path at ',num2str(time),' seconds']);
xlim([0 256]);
ylim([0 360]);
xticks(0:50:256)
yticks(0:30:360)
ax = gca;
ax.YDir = 'normal';
view(0,90);
hold on
mesh(Sfout(:,:,ts));
hold off
%% Irradiation matrix on the solar field
time=t(ts);
imagesc(Irrfield(:,:,ts));
grid on
xlabel('Abscissa direction (m)');
ylabel('North direction (m)');
xlim([0 256]);
ylim([0 360]);
xticks(0:50:256)
yticks(0:30:360)
ax = gca;
ax.YDir = 'normal';
title(['Irradiation at ',num2str(time),' seconds'])
colorbar
c = colorbar;
c.Label.String = 'W/m²';
map=[0 1 1
    0 0 1
    1 1 0
    1 0 1
    1 0 0];
colormap(map)

%% Irradiation on the solar field
time=t(ts);
imagesc(irr_Tf(:,:,ts));
grid on
xlabel('Abscissa direction (m)');
ylabel('North direction (m)');
xlim([0 256]);
ylim([0 360]);
xticks(0:50:256)
yticks(0:30:360)
ax = gca;
```

```
ax.YDir = 'normal';
title(['Irradiation on modules at ',num2str(time),' seconds'])
colorbar
c = colorbar;
c.Label.String = 'W/m²';
map=[1 1 1
    0 0 1
    1 1 0
    1 0 1
    1 0 0];
colormap(map)

%% Module temperature in the in the trakcer layout
Tamb=-3.75;%°C
NOCT=43;%°C
Tplant=(irr_Tf/800)*(NOCT-20)+Tamb;
ts=2001;
time=t(ts);
imagesc(Tplant(:,:,ts));
grid on
xlabel('Module witdh (m)');
ylabel('string length (m)');
%axis equal
xlim([0 256]);
ylim([0 360]);
xticks(0:50:256)
yticks(0:30:360)
ax = gca;
ax.YDir = 'normal';
title(['Module temperature at ',num2str(time),' seconds'])
colorbar
caxis([0 30]);
c = colorbar;
c.Label.String = '°C';
colormap(flipud(hot));

%% Generating matrixes for the ouput powers !!! (this part has not connection with the
ploted graphs but it is used to construct the outputs of the PV modules)
t_Modulefield=size(t_Modulefield_v2);
deci=10;%decimation to filter values in case the matrixes of time are really big
t_rel=floor(t_Modulefield(1)/deci);% rounding down so as not to crack the simulation
[a,b,c]=size(irr_Tf);
P_out=zeros(a,b,t_rel);
I_out=zeros(a,b,t_rel);
t_out=zeros(1,t_rel);
for t_M=1:1:t_rel
    n=1;
    string=1;
    mm=0;
    t_aux=t_M*deci;
    t_out(1,t_M)=t_Modulefield_v2(t_aux);
```

```
  while n<b
     suma=0;
     m=0;
       while m<60
       m=m+1;
       mm=mm+1;
       t_mes=t_M*deci;
       P_out(mm,n,t_M)=P(t_mes,string);
       I_out(mm,n,t_M)=I(t_mes,string);
          if m==60
             string=string+1;
          end
          if mm==a
             n=n+5;
             mm=0;
          end
       end
  end
end

%% Power generation of the PV modules
time=t_Modulefield_v2(ts2*10);
imagesc(P_out(:,:,ts2));
grid on
xlabel('Abscissa direction (m)');
ylabel('North direction (m)');
xlim([0 256]);
ylim([0 360]);
xticks(0:50:256)
yticks(0:30:360)
ax = gca;
ax.YDir = 'normal';
title(['Power production per string at ',num2str(time),' seconds'])
colorbar
c = colorbar;
c.Label.String = 'W';
colormap(flipud(hot));

%% MPPT power prodution
subplot(3,1,1)
plot(t_Modulefield_v2,PpvPiO/1000);
grid on
xlabel('time (s)');
ylabel('P (kW)');
xlim([0 25]);
ylim([0 3000]);
title('Power plant production');
subplot(3,1,2)
plot(t_Modulefield_v2,VpvPiO);
xlabel('time (s)');
ylabel('Voltage (V)');
```

```
xlim([0 25]);
ylim([0 1300]);
title('MPPT DC Voltage');
subplot(3,1,3)
plot(t_Modulefield_v2,IpvPiO);
xlabel('time (s)');
ylabel('Current (A)');
xlim([0 25]);
ylim([1500 2700]);
title('MPPT DC Current');

%% I-V Curve at a constant irradiation (t=14 seconds)
plot(V_IVCurve,I_IVCurve);
grid on
xlabel('Voltage (V)');
ylabel('Current (A)');
%axis equal
%xlim([0.5 1.5]);
%ylim([0 60]);
%xticks(1:1:1);
%yticks(0:2:60);
yyaxis left
title(['I-V Curve at 14 seconds'])
yyaxis right
plot(V_IVCurve,P_IVCurve/1000);
legend('I-V Curve','P-V Curve')

%% Overall energy produced along the simulation per each string
imagesc(E_out(:,:,10014))
grid on
xlabel('Abscissa direction (m)');
ylabel('North direction (m)');
xlim([0 256]);
ylim([0 360]);
xticks(0:50:256)
yticks(0:30:360)
ax = gca;
ax.YDir = 'normal';
title(['Overall energy produced per each string'])
colorbar
caxis([0 60]);
c = colorbar;
c.Label.String = 'Wh';
colormap(flipud(hot));

%% Detailed power generation for a given string
ts=2001;
time=t(ts);
subplot(1,2,1)
imagesc(P_outstring(:,ts));
grid on
```

```
xlabel('Module witdh (m)');
ylabel('string length (m)');
axis equal
xlim([0.5 1.5]);
ylim([0 60]);
xticks(1:1:1)
yticks(0:2:60)
%ax = gca;
%ax.YDir = 'normal';
title(['Module Power production at ',num2str(time),' seconds'])
colorbar
caxis([0 330]);
c = colorbar;
c.Label.String = 'W';
subplot(1,2,2)
imagesc(Irrstring(:,:,ts));
grid on
xlabel('Module witdh (m)');
ylabel('string length (m)');
axis equal
xlim([0.5 1.5]);
ylim([0 60]);
xticks(0:1:1)
yticks(0:2:60)
%ax = gca;
%ax.YDir = 'normal';
title(['Module Irradiation at ',num2str(time),' seconds'])
colorbar
caxis([0 800]);
c = colorbar;
c.Label.String = 'W/m²';
%% Detailed irradiaton string
ts=2001;
time=t(ts);
imagesc(Irrstring(:,:,ts));
grid on
xlabel('Module witdh (m)');
ylabel('string length (m)');
axis equal
xlim([0.5 1.5]);
ylim([0 60]);
xticks(0:1:1)
yticks(0:2:60)
%ax = gca;
%ax.YDir = 'normal';
title(['Module Irradiation at ',num2str(time),' seconds'])
colorbar
caxis([0 800]);
c = colorbar;
c.Label.String = 'W/m²';
```

```
%% Detailed string MPPT power prodution
grid on
subplot(3,1,1)
plot(t_DetailedString,Pmpp_string);
grid on
xlabel('time (s)');
ylabel('P (W)');
xlim([0 25]);
ylim([0 10000]);
title('String power production');
subplot(3,1,2)
plot(t_DetailedString,Pmpp_string./Impp_string);
grid on
xlabel('time (s)');
ylabel('Voltage (V)');
xlim([0 25]);
ylim([0 1300]);
title('String Voltage');
subplot(3,1,3)
plot(t_DetailedString,Impp_string);
grid on
xlabel('time (s)');
ylabel('Current (I)');
xlim([0 25]);
ylim([0 10]);
title('String Current');

%% Moldule termperature evaluation in a given string
Tamb=-3.75;%°C
NOCT=43;%°C
Tstring=(Irrstring/800)*(NOCT-20)+Tamb;
subplot(1,2,1)
ts=1401;
time=t(ts);
imagesc(Tstring(:,:,ts));
grid on
xlabel('Module witdh (m)');
ylabel('string length (m)');
axis equal
xlim([0.5 1.5]);
ylim([0 60]);
xticks(0:1:1)
yticks(0:2:60)
title(['Module temperature at ',num2str(time),' seconds'])
colorbar
caxis([0 30]);
c = colorbar;
c.Label.String = '°C';
subplot(1,2,2)
ts=2001;
time=t(ts);
```

```
imagesc(Tstring(:,:,ts));
grid on
xlabel('Module witdh (m)');
ylabel('string length (m)');
axis equal
xlim([0.5 1.5]);
ylim([0 60]);
xticks(0:1:1)
yticks(0:2:60)
title(['Module temperature at ',num2str(time),' seconds'])
colorbar
caxis([0 30]);
c = colorbar;
c.Label.String = '°C';

%% Overall energy produced per each module in a given string
imagesc(E_outstring(:,2523))
grid on
xlabel('Module witdh (m)');
ylabel('string length (m)');
ax = gca;
ax.YDir = 'normal';
axis equal
xlim([0.5 1.5]);
ylim([0 60]);
xticks(1:1:1)
yticks(0:2:60)
title(['Cumulative energy produced per each module'])
colorbar
caxis([0 2]);
c = colorbar;
c.Label.String = 'Wh';
colormap(flipud(hot));

%% InverterGrid Voltage q
plot(t_invertergrid,Vzq,'b');
grid on
xlabel('time (s)');
ylabel('Voltage (V)');
xlim([0 24]);
ylim([700 900]);
title('Vzq and Vlq');
hold on
plot(t_invertergrid,Vlq,'r');
legend('Vzq','Vlq')
hold off

%% InverterGrid Voltage q (zoom in)
subplot(3,1,1)
plot(t_invertergrid,Vzq,'b');
grid on
```

```
xlabel('time (s)');
ylabel('Voltage (V)');
xlim([0 0.2]);
ylim([850 970]);
title('Vzq and Vlq');
hold on
plot(t_invertergrid,Vlq,'r');
legend('Vzq','Vlq')
hold off
subplot(3,1,2)
plot(t_invertergrid,Vzq,'b');
grid on
xlabel('time (s)');
ylabel('Voltage (V)');
xlim([1.98 2.02]);
ylim([866 867.5]);
title('Vzq and Vlq');
hold on
plot(t_invertergrid,Vlq,'r');
legend('Vzq','Vlq')
hold off
subplot(3,1,3)
plot(t_invertergrid,Vzq,'b');
grid on
xlabel('time (s)');
ylabel('Voltage (V)');
xlim([10 10.2]);
ylim([866 867.5]);
title('Vzq and Vlq');
hold on
plot(t_invertergrid,Vlq,'r');
legend('Vzq','Vlq')
hold off

%% InverterGrid Voltage d
plot(t_invertergrid,Vzd,'b');
grid on
xlabel('time (s)');
ylabel('Voltage (V)');
xlim([0 24]);
%ylim([-5 5]);
title('Vzd and Vld');
hold on
plot(t_invertergrid,Vld,'r');
legend('Vzd','Vld')
hold off

%% InverterGrid Voltage d (zoom in)
subplot(2,1,1)
plot(t_invertergrid,Vzd,'b');
grid on
```

```
xlabel('time (s)');
ylabel('Voltage (V)');
xlim([0 2]);
ylim([-10 0.5]);
title('Vzd and Vld');
hold on
plot(t_invertergrid,Vld,'r');
legend('Vzd','Vld')
hold off
subplot(2,1,2)
plot(t_invertergrid,Vzd,'b');
grid on
xlabel('time (s)');
ylabel('Voltage (V)');
xlim([15 15.2]);
ylim([-10 0.5]);
title('Vzd and Vld');
hold on
plot(t_invertergrid,Vld,'r');
legend('Vzd','Vld')
hold off
%% Voltages on the abc frame
subplot(2,1,1)
plot(t_invertergrid,Vzabc);
grid on
xlabel('time (s)');
ylabel('Voltage (V)');
xlim([0 0.4]);
ylim([-1000 1000]);
title('Vzabc');
legend('Va','Vb','Vc')
subplot(2,1,2)
plot(t_invertergrid,Vlabc);
grid on
xlabel('time (s)');
ylabel('Voltage (V)');
xlim([0 0.4]);
ylim([-1000 1000]);
title('Vlabc');
legend('Va','Vb','Vc')

%% Frequency response
plot(t_invertergrid,f);
grid on
xlabel('time (s)');
ylabel('Frequency (hz)');
xlim([0 24]);
ylim([0 60]);
title('Frequency response');
legend('frequency')
```

```
%% Q and D currents
subplot(2,1,1)
plot(t_invertergrid,Iq);
grid on
xlabel('time (s)');
ylabel('current (A)');
xlim([0 24]);
%ylim([0 60]);
hold on
plot(t_invertergrid,Iqref);
legend('iq','iq*');
hold off
subplot(2,1,2)
plot(t_invertergrid,Id);
grid on
xlabel('time (s)');
ylabel('current (A)');
xlim([0 24]);
%ylim([0 60]);
hold on
plot(t_invertergrid,Idref);
legend('id','id*');
hold off

%% Q and D currents (zoom in)
subplot(4,1,1)
plot(t_invertergrid,Iq);
grid on
xlabel('time (s)');
ylabel('current (A)');
xlim([0 0.6]);
ylim([-10000 2000]);
hold on
plot(t_invertergrid,Iqref);
legend('iq','iq*');
hold off
subplot(4,1,2)
plot(t_invertergrid,Iq);
grid on
xlabel('time (s)');
ylabel('current (A)');
xlim([15 15.02]);
ylim([-1700 -1500]);
hold on
plot(t_invertergrid,Iqref);
legend('iq','iq*');
hold off
subplot(4,1,3)
plot(t_invertergrid,Id);
grid on
xlabel('time (s)');
```

```
ylabel('current (A)');
xlim([0.098 0.13]);
ylim([-80 20]);
hold on
plot(t_invertergrid,Idref);
legend('id','id*');
hold off
subplot(4,1,4)
plot(t_invertergrid,Id);
grid on
xlabel('time (s)');
ylabel('current (A)');
xlim([15 15.02]);
ylim([-1 1]);
hold on
plot(t_invertergrid,Idref);
legend('id','id*');
hold off

%% abc currents
subplot(4,1,1)
plot(t_invertergrid,Izabc);
grid on
xlabel('time (s)');
ylabel('current (A)');
xlim([0.4 0.8]);
%ylim([0 60]);
legend('Ia','Ib','Ic');
subplot(4,1,2)
plot(t_invertergrid,Izabc);
grid on
xlabel('time (s)');
ylabel('current (A)');
xlim([1 1.5]);
%ylim([0 60]);
legend('Ia','Ib','Ic');
subplot(4,1,3)
plot(t_invertergrid,Izabc);
grid on
xlabel('time (s)');
ylabel('current (A)');
xlim([12 12.4]);
%ylim([0 60]);
legend('Ia','Ib','Ic');
subplot(4,1,4)
plot(t_invertergrid,Izabc);
grid on
xlabel('time (s)');
ylabel('current (A)');
xlim([21 21.4]);
%ylim([0 60]);
```

```
legend('Ia','Ib','Ic');


%% E_DC
plot(t_invertergrid,E_DC);
grid on
xlabel('time (s)');
ylabel('Voltage (V)');
xlim([0 24]);
%ylim([0 60]);
legend('E_D_C');

%% Converted AC power
hold off
plot(t_invertergrid,-P_AC/1000,'r');
grid on
xlabel('time (s)');
ylabel('Power (kW)');
xlim([0 24]);
ylim([0 3000]);
hold on
plot(t_Modulefield_v2,PpvPiO/1000,'b');
legend('DC Power','AC Power');
hold off
```